

LIBRARY OF THE
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

510.84

Il6t

1972





Digitized by the Internet Archive
in 2013

QUARTERLY TECHNICAL PROGRESS REPORT

January, February, March 1972

Closet 2B-15

THE LIBRARY OF THE
NOV 27 1972
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

151
QUARTERLY TECHNICAL PROGRESS REPORT

January, February, March 1972

UIUCDCS-QPR-72-1

TABLE OF CONTENTS

	Page
1. CIRCUIT RESEARCH.	1
1.1 Bundle Processing (Project No. 21)	2
1.1.1 BURP (formerly known as Bundle Repeater-Restorer).	2
1.2 APE (Project No. 25)	5
1.3 PENTECOST (Project No. 31)	7
1.4 Ergodic (Project No. 39)	7
1.4.1 Summary	7
1.4.2 Ergodic Generator	9
1.4.3 Processor	9
1.5 Telemaze (Project No. 41)	14
1.5.1 Video Generator Progress.	14
1.5.1.1 Goal	14
1.5.1.2 Progress Summary	14
1.5.1.3 Block Diagram.	14
1.5.2 Video Square Generator.	17
1.5.2.1 Objective.	17
1.5.2.2 Block Diagram.	17
2. HARDWARE SYSTEMS RESEARCH	20
2.1 LASCOT (Project No. 09).	21
2.2 OLFT (Project No. 12).	21
2.2.1 Temperature Sense and Control Circuits.	21
2.2.2 Photon Couplers	23
2.3 LINDA (Project No. 28)	28
2.3.1 Summary	28
2.3.2 Project Status.	28
2.3.2.1 1-Line Video Delay	30
2.3.2.2 Separation Logic	30
2.3.3 Future Work	32
2.4 RASER (Project No. 29).	32
2.5 Stereomatrix (Project No. 30).	33
2.5.1 Coefficient Generator	33
2.5.2 Inverse Transformer and Cursor.	33
2.5.3 Display	40
2.5.4 Coefficient Generator Redesign.	40
2.6 Scantrix (Project No. 35).	43
2.6.1 TV Signals.	43
2.6.2 Composite Video	43
2.6.3 Horizontal Sync	46
2.6.4 Vertical Sync	46
2.6.5 Phase-Locked Oscillator (System Clock).	46

	Page
2.7	Frog (Project No. 36) 49
2.7.1	Project Status 49
2.7.2	The World Model: Structure of the Memory 49
2.7.3	The World Model: Memory Integration Unit 51
2.8	BEACON (Project No. 38) 53
2.8.1	Introduction 53
2.8.2	Work Done 53
2.8.3	Alternative Method 53
2.8.4	Future Work 53
3.	SOFTWARE SYSTEMS RESEARCH 57
3.1	Numerical Processes 57
3.1.1	Ordinary Differential Equations 57
3.1.2	Sparse Matrix Inversion 58
3.1.3	The Steady-State Package 59
3.1.4	Plot Package 73
3.1.5	Matrix Inversion Package 77
	3.1.5.1 Modifications to SPACT 77
	3.1.5.2 Changes to ITEM and COG 87
3.2	Illinois Graphics Computing System 94
3.2.1	Mini-Machine Modeling System 102
3.3	Graphical Remote Access Support System 104
3.3.1	Disk Monitor System 104
3.3.2	Information Retrieval 106
3.3.3	Monitors 107
3.3.4	Remote Data Structure Utilities 109
3.4	Computer Maintenance and Construction 111
3.4.1	Graphics-8 Hardware 111
3.4.2	Equipment Maintenance Log Summary 112
4.	IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH:
ILLIAC III 115
4.1	Introduction 115
4.2	Interactive Picture Processing 118
4.2.1	Show-and-Tell 118
4.2.2	Atlas Image Deformation 118
4.2.3	Scan/Display Device Development 125
4.3	Parallel Processing Strategies 126
4.3.1	Texture Analysis 126
4.3.2	Covering Theory 126
4.3.3	Variable-valued Logic 126
4.3.4	PAX Language Support 127
4.4	Structural Inference 128
4.4.1	Scene Segmentation 128
4.4.2	Shape Identification 129

4.4.3	Structure Transformations	130
4.4.4	SOL (Structure Operation Language).	132
4.5	Applications	133
4.5.1	Cervical Smears	133
4.5.2	Brain Mapping	133
4.5.3	Cytospectrometer.	134
4.6	Computer Systems Support	135
4.6.1	IBAL Assembler.	135
4.6.2	PAU Development	135
4.6.3	Processor Intercommunication.	135
4.6.4	Scanner/Monitor	135
4.7	BIBLIOGRAPHY	138
4.7.1	Outside Lectures.	138
4.7.2	Logic Drawings Issued	138
4.7.3	Engineering Drafting Report	138
4.8	ADMINISTRATION	139
4.8.1	Personnel Report.	139
5.	CENTER FOR ADVANCED COMPUTATION	140
5.1	ALGORITHM DEVELOPMENT GROUP.	142
5.1.1	Introduction.	142
5.1.2	Computational Methods in Linear Algebra	143
5.1.2.1	Solution of Systems of Linear Equations	143
5.1.2.2	The Algebraic Eigenvalue Problem.	144
5.1.3	Linear Programming.	146
5.1.4	Ordinary and Partial Differential Equations	147
5.1.5	Time-Series Analysis.	150
5.1.6	QUASCO: An ILLIAC IV Audratic Assignment Code	150
5.1.7	Graphics Algorithm Development for ILLIAC IV	152
5.1.7.1	Surface Presentation of a Function F(X,Y).	152
5.1.7.2	Contour Plotting Algorithm for ILLIAC IV.	153
5.1.8	Miscellaneous	153
5.2	NETWORK SYSTEMS AND SERVICES GROUP	155
5.2.1	Introduction.	155
5.2.2	ARPA Network Activities	155
5.2.2.1	ARPA Network Terminal System (ANTS) Development..	155
5.2.2.2	ARPA Network Usage	156
5.2.2.3	Further Installations of ANTS Systems on the Network.	156

	Page
5.2.3 Graphics Support for Center Projects on the B6700.	157
5.2.4 Network Graphics Efforts.	158
5.2.4.1 Network Graphics Protocol..	158
5.2.4.2 Laboratory for Atmospheric Research (LAR) Support. . .	158
5.3 ECONOMIC RESEARCH GROUP.	159
5.4 ILLIAC IV LANGUAGE DEVELOPMENT FOR THE PHASE II SYSTEM.	161
5.4.1 Summary	161
5.5 ILLIAC IV INFORMATION RETRIEVAL AND STATISTICAL SYSTEM (IRSS).	162
5.5.1 Summary	162
5.6 ILLIAC IV IMAGE PROCESSING SYSTEM.	164
5.6.1 Summary	164
5.7 EDUCATIONAL ACTIVITIES	165
5.7.1 Documentation	165
5.7.2 Teaching and Consulting	165
5.7.3 Training.	165
5.8 NARIS.	166
5.9 IRIS	167
6. THEORY OF DIGITAL COMPUTER ARITHMETIC.	173
7. SWITCHING THEORY AND LOGICAL DESIGN	174
8. OFFICE OF THE SOUPAC CONSULTANTS.	177
9. MACHINE AND SOFTWARE ORGANIZATION STUDIES	179
9.1 FORTRAN Parallelism Detection.	179
9.2 Preliminary Results.	180
9.3 Simulation Processor	183
9.4 Text Searching	183
9.5 S-Level Program.	184
9.6 Hardware Support	184
9.7 D-Machine Assembler.	185
9.8 SPITBOL.	186
10. COMPUTER SYSTEMS ANALYSIS	187
10.1 Computer Network Modeling.	187
10.2 Center Throughput Analysis	187
11. NUMERICAL ANALYSIS.	191
11.1 Extensions of Factorization Methods and Adaptive Algorithms.	191

11.2	Eigenvalue Bounds for the Iteration Matrix of Stone's Factorization	192
11.3	A Fast Direct Method for the Solution of Linear Equations	194

12.	NUMERICAL METHODS, COMPUTER ARITHMETIC AND ARTIFICIAL LANGUAGES	195
12.1	MIPS	195
12.2	GIZMO.	198
	12.2.1 Introduction.	198
	12.2.2 What GIZMO is Now	198
	12.2.3 Near Future Plans	199
	12.2.4 Long-Range Goals.	200
	12.2.5 So what's so new about GIZMO?	200
	12.2.6 Conclusions	201
12.3	Appendix A	203
	12.3.1 Teacher's Manual for Teacher Mode of GIZMO	203
12.4	PAL-11 Source Compression.	212
12.5	ETS.	213
12.6	Magnetic Tape System for ETS (MAGETS).	214
13.	GENERAL DEPARTMENT INFORMATION.	217
13.1	Personnel.	217
13.2	Bibliography	218
13.3	Colloquia.	220
13.4	Drafting	221
13.5	Shop's Production.	221

1. CIRCUIT RESEARCH

(Supported in part by the Office of Naval Research under Contract
N000 14-67-A-0305-0007, W. J. Poppelbaum, Principal Investigator.)

Summary

Bernard Tse's Bundle Repeater/Restorer project is now called BURP¹. His report describes a pair of selector circuit designs. Yiu Wo's APE report contains a control unit block diagram and a preliminary test account of a prototype APE. Panigrahi has designed the high voltage switching circuits for PENTECOST. Also in the stochastic area, Jim Cutler describes the design of an "Ergodic" generator for the project of the same name. An ergodic bundle is one for which the space and time averages are equal, allowing an increased measure of system redundancy. Finally, Ed Pott presents an account of the video processing involved in the Telemaze project.

M. Faiman - editor

[¹The editor disclaims all responsibility for project names.]

1.1 Bundle Processing (Project No. 21)

1.1.1 BURP (formerly known as Bundle Repeater-Restorer)

Figures 1 and 2 show the diagrams and truth tables for the two types of selectors of BURP. The inputs carry the binary signal levels "1" and "0" represented by +5V and -5V, respectively. The outputs carry the ternary levels "+", "-" and "0" represented by +5V, -5V and 0V, respectively. These two circuits are part of the restorer of BURP.

The repeater circuit cards have been tested and fabrication of these cards is underway. The first repeater section should be completed soon.

Bernard Tse

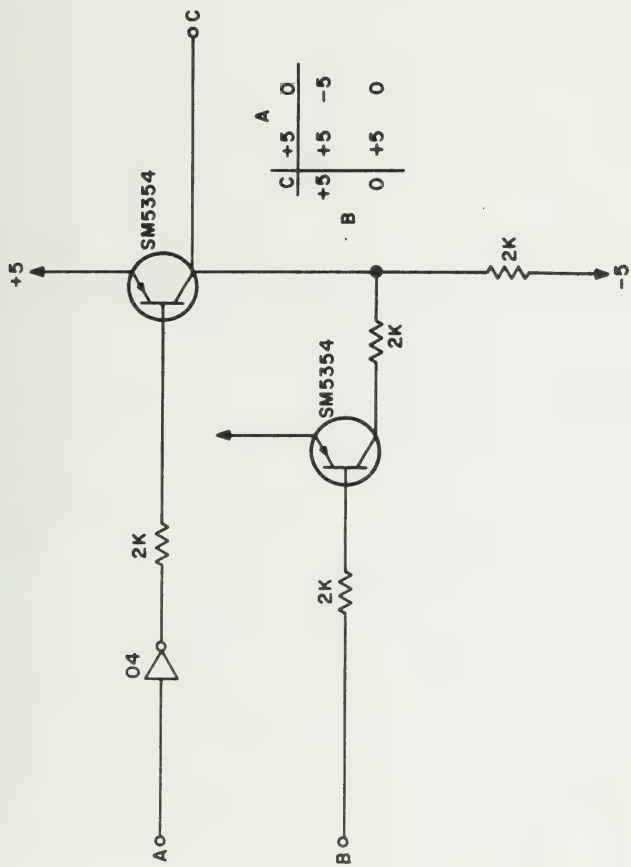


Figure 1. Selector A of Restorer

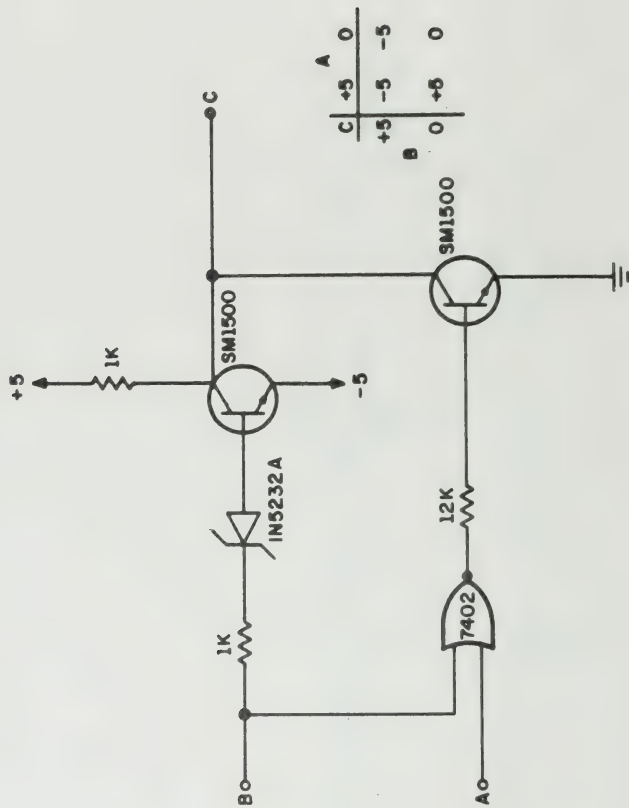


Figure 2. Selector B of Restorer

1.2 APE (Project No. 25)

During the past quarter, the design of the APE control unit has been completed. All the required circuits have been built and the control unit is now being assembled and debugged. It is expected that the control unit will be in full operation in the following quarter. As described in the previous reports, the control unit has two major functions. It transmits the tuning and operational instructions to all APEs. According to these instructions the APEs organize themselves into a specific tree structure to perform a specific type of program. Furthermore, the control unit tests all the APEs involved and displays their outputs. A block diagram of the control unit is shown in Figure 1. Details of many functional blocks in this diagram have been described in the previous reports. The remaining ones will be reported on in a special report for the APE machine.

In addition to the progress with the control unit, a prototype APE has been built during this quarter. It is now under testing and evaluation. Preliminary data shows that it meets all major requirements of the APE machine. In particular, its total power consumption is well within the 100 mW limit.

Yiu Wo

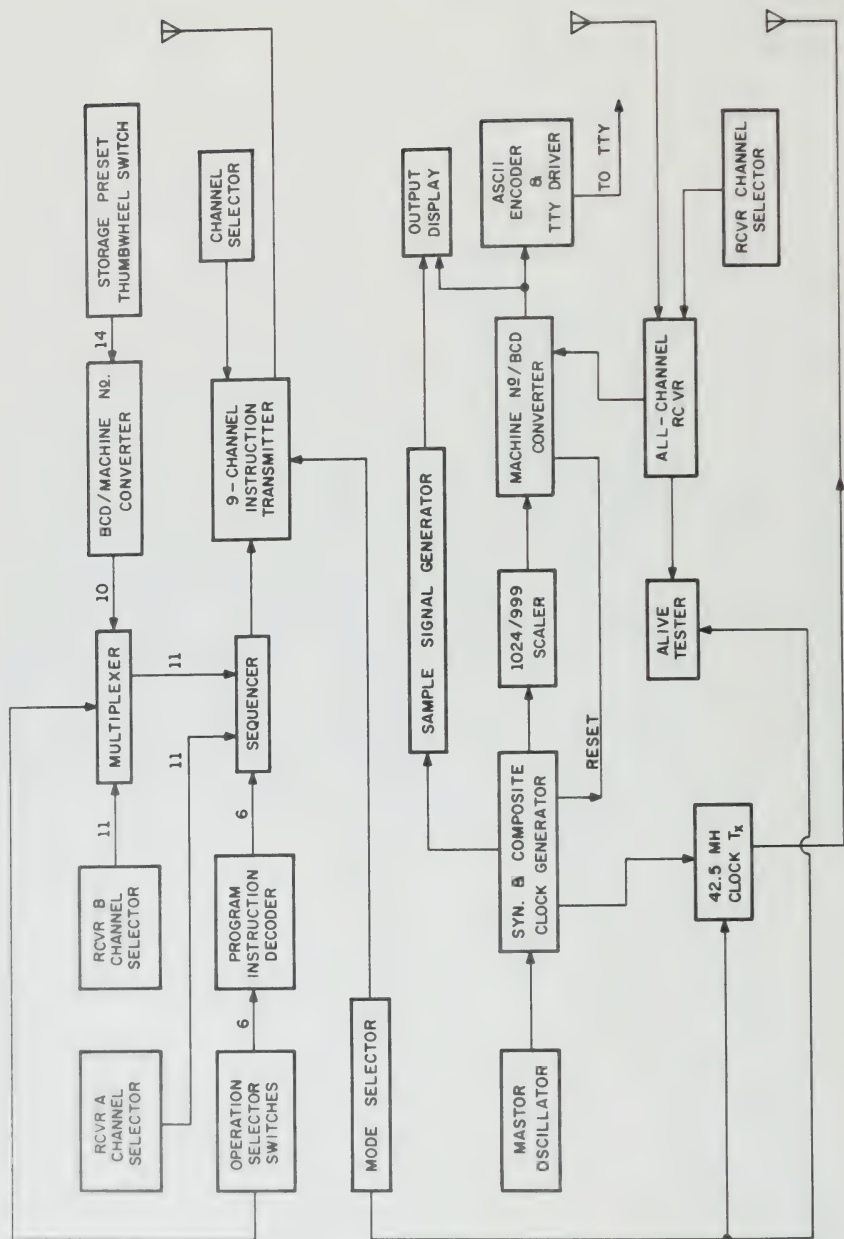


Figure 1. A Block Diagram of the APE Control Unit

1.3 PENTECOST (Project No. 31)

A high voltage switch has been assembled and tested to switch the anode voltage from 16kV to 11kV. A similar switch switches the focus voltage from 2.4kV to 1.6kV. The size control circuits are being designed to switch the vertical and horizontal sizes. Some preliminary tests are planned with the filter wheel rotating in front of the camera and the screen voltage being switched synchronously with it. It is planned that a photocircuit would detect the position of the filter wheel and a signal derived from this would be used to synchronize the camera and hence, consequently the monitor.

G. Panigrahi

1.4 Ergodic (Project No. 39)

1.4.1 Summary

An ergodic generator will represent a number using a bundle of wires by two methods. First, the number will be represented by the number of wires that are energized at any one time and second, the same number will be represented by the number of times any particular wire is energized over a period of time. This redundancy allows one to check for malfunctions in the circuitry. Arithmetic operations can be performed using two ergodic bundles (derived from ergodic generators). The output from the arithmetic unit will be another ergodic bundle which can be decoded into a numerical solution and checked because of the redundancy of an ergodic bundle. Figure 1 shows the block diagram of the above description.

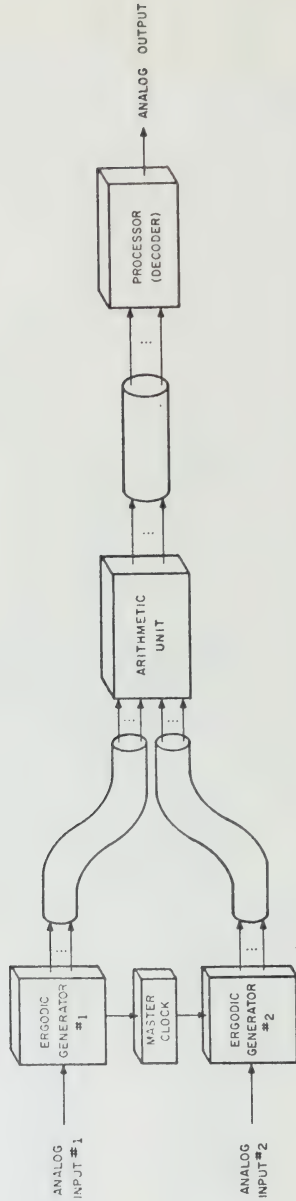


Figure 1. Ergodic Block Diagram

In designing this system there are two desirable properties:

- 1) The two ergodic generators should be independent of each other.

This property is necessary in order to obtain the correct answer from the arithmetic generator.

- 2) The output of the arithmetic unit should be an ergodic bundle as noted above.

1.4.2 Ergodic Generator

A card has been built by the electronics shop which will be used for making an ergodic generator. It is shown in Figure 2. A reasonable number of wires for a bundle has been chosen to be 64 so that two of the cards will be necessary to make one ergodic generator. The two cards will be connected in such a way that the generator will be a 64 -bit circular shift register. The technique used to build two ergodic generators that are independent will be discussed in the next quarterly report.

1.4.3 Processor

The processor (or decoder) is under construction. Two of the required four cards are completed: the 32 -bit shift register and the timing control cards. The time average and the comparison cards are in the electronics shop at this time. The circuit diagrams are shown in Figures 2 through 5. The purpose of this unit is to count the number of times a wire is energized during a unit of time (the unit of time depends upon the frequency of the master clock in Figure 1) for a period of time, τ , and to compare this value with the number of energized wires at a certain time. In other words the processor compares a time average of a wire being energized with a space average of the bundle of wires being energized. The processor will make this comparison for each wire and thus can determine if the wire and the circuitry associated

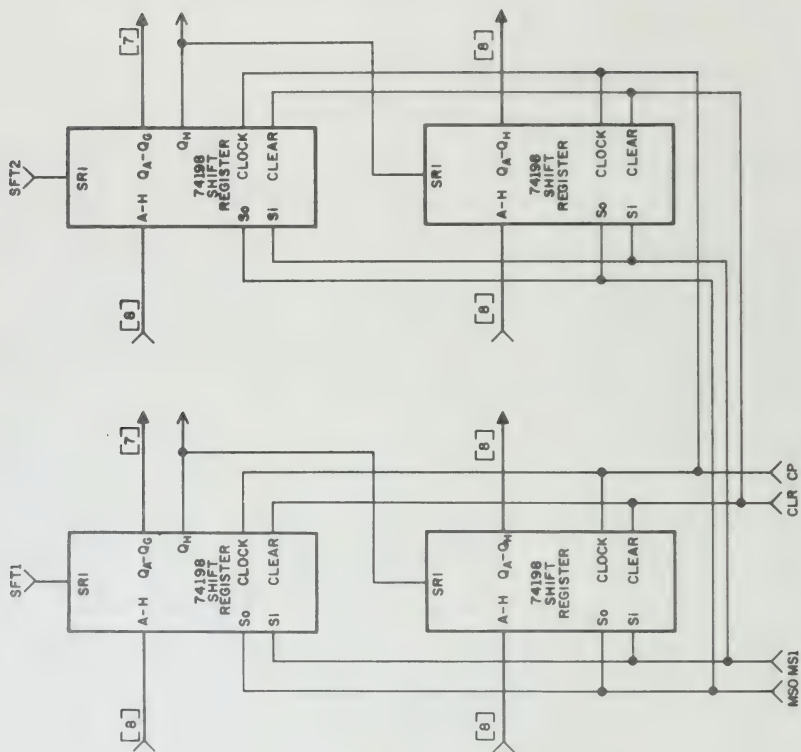


Figure 2. 32-bit Shift Register Card

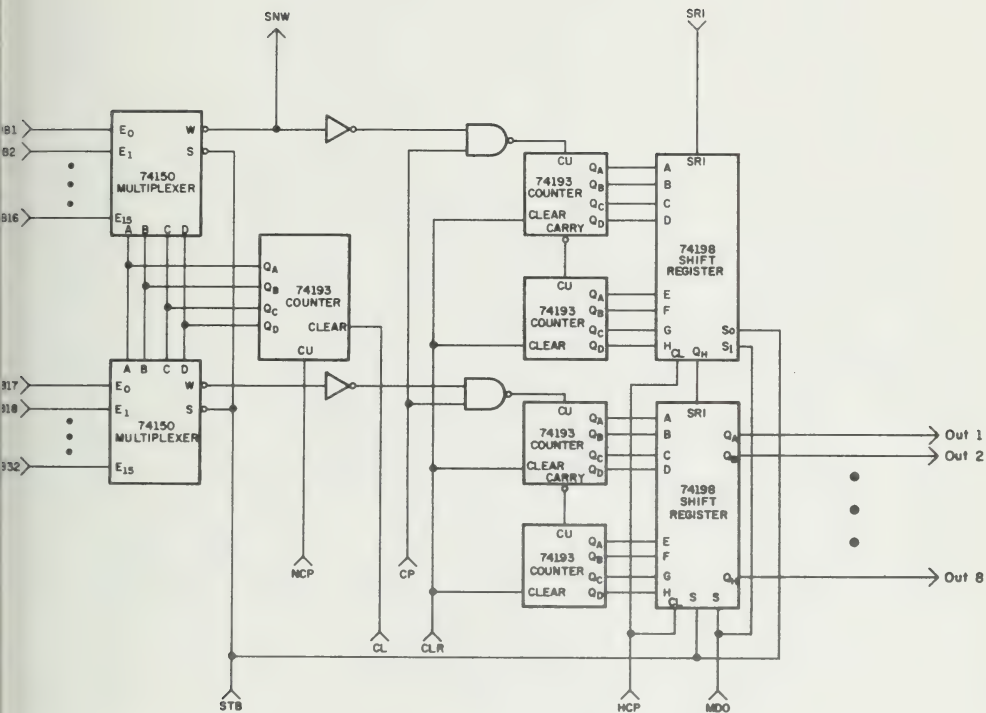


Figure 3. Time Average Card

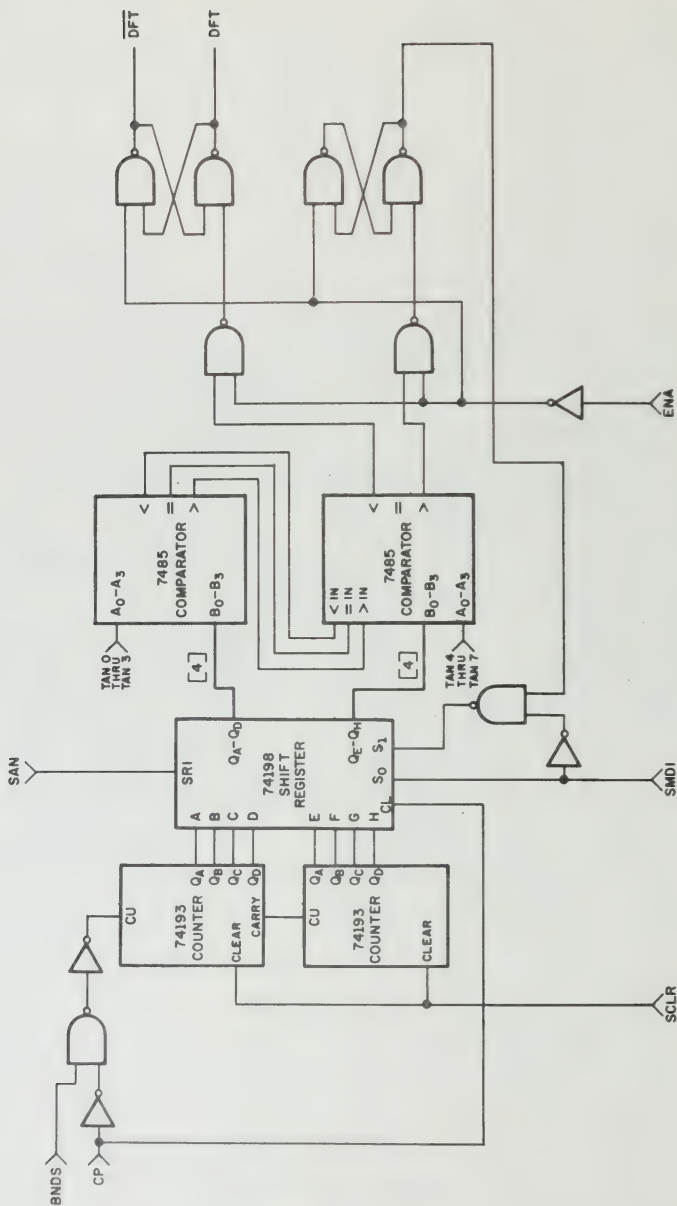


Figure 4. Comparison Card

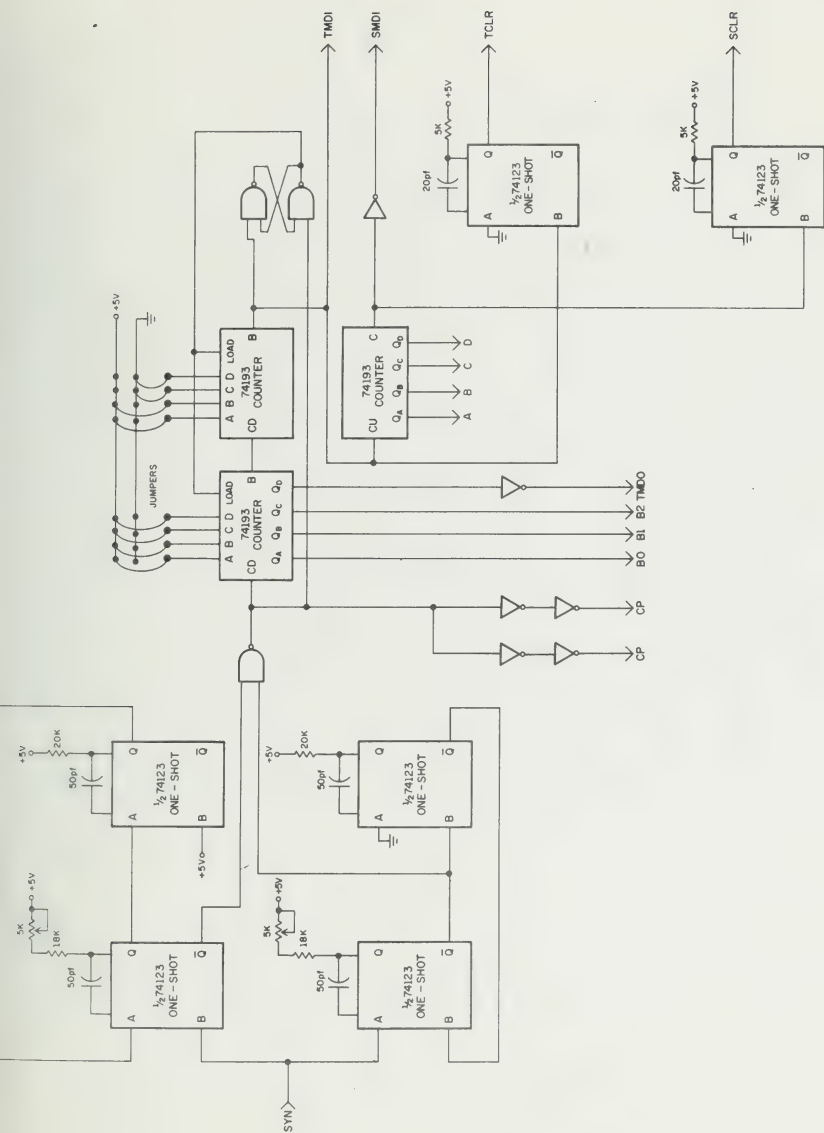


Figure 5. Timing Control Card

with that wire are still operating properly. It will also update the space average every four periods, 4 τ .

Jim Cutler

1.5 Telemaze (Project No. 41)

1.5.1 Video Generation Progress

1.5.1.1 Goal

The basic function of the video generator is to produce the video presentation on a television monitor cathode ray tube (CRT). It consists of a pattern of 256 squares in a 16 row by 16 column arrangement. This is shown in Figure 1. The matrix itself should be square so that the pattern displayed appears as smaller squares. The standard television CRT face is rectangular, and therefore some of the picture area must be left at the black level. This unused portion is represented by the shaded portion of the video.

1.5.1.2 Progress Summary

The video generated is to be displayed on a standard television monitor. A 23-inch black and white monitor has been selected and is being evaluated. Standard synchronizing signals have been obtained from duplicating the sync generator card designed for RASER¹. It was noted that the sync card and the monitor could be interconnected directly. This avoids adding a negative power supply to the system. This was a result of the monitor having capacitively coupled inputs and a DC restorer.

1.5.1.3 Block Diagram

The major components needed to generate the pattern previously discussed are shown in Figure 2. The present method investigated utilized a high frequency clock which is an integral number of cycles of the television

¹The RASER (Project No. 29) sync card was explained in the previous report.

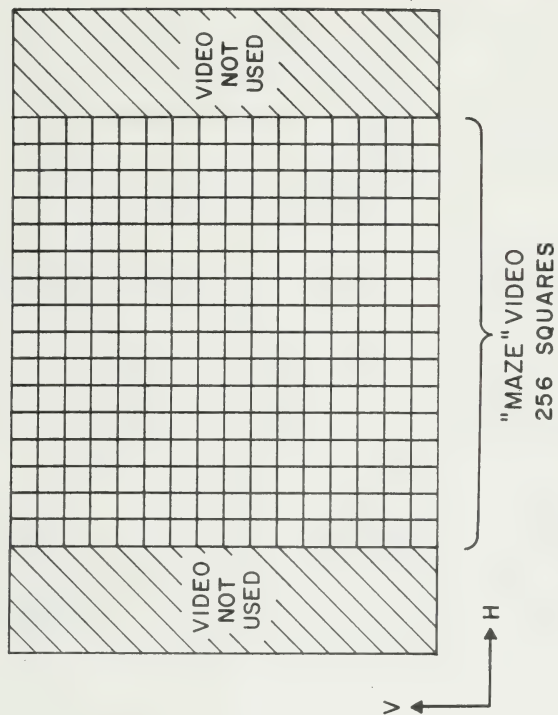


Figure 1. Television CRT Face (Active Video Area)

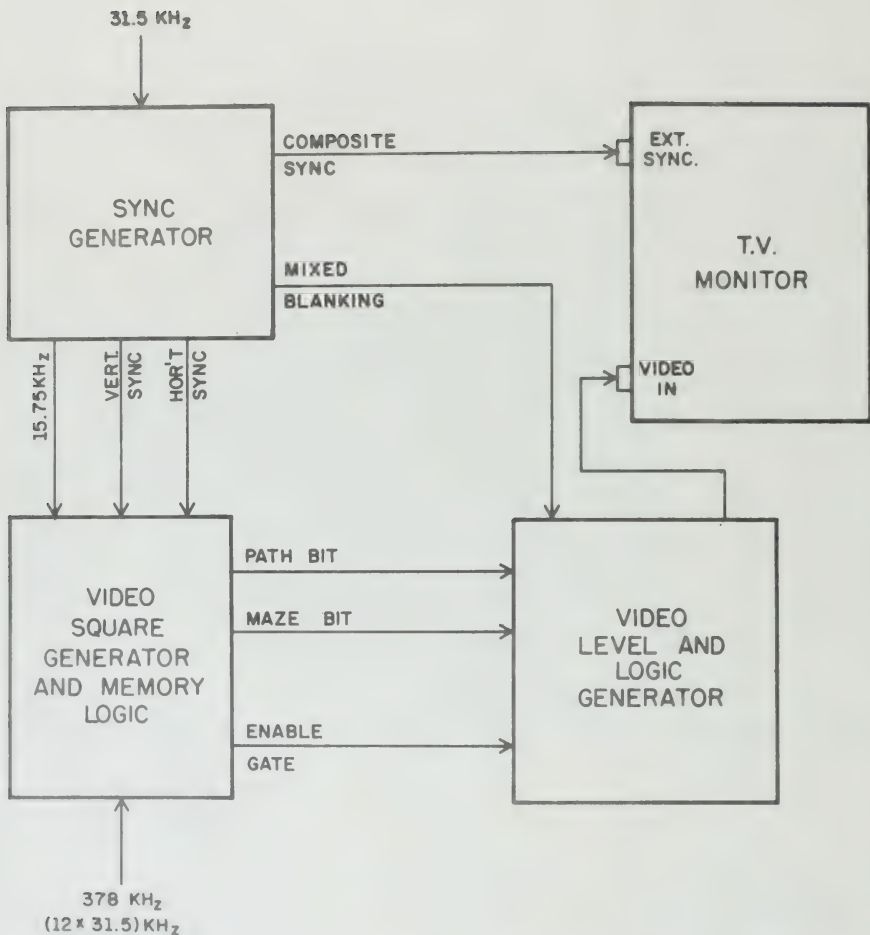


Figure 2. Video Block Diagram (Local Station)

clock. The period of this high frequency clock cycle is then used to determine the horizontal length of video square on the CRT. Also the sync generator supplies the necessary sync signals which allow the addressing logic to determine which is to be enabled.

The video square generator and memory logic block contains the signals necessary to address both the maze and path generator. It also generates the video enable gate which enables the video logic generator to process the memory bits. The video level generator uses the 2 bits of serial information and establishes 4 levels of video for the CRT.

1.5.2 Video Square Generator

1.5.2.1 Objective

This logic block has to generate the timing signals necessary for displaying the contents in the Model maze and path 256 bit memories. It must convert 16 bits per row into synchronized serial information for the television presentation.

1.5.2.2 Block Diagram

The synchronization needed between the television monitor and the video square generator is obtained from the sync generator card (Figure 3). The sync generator used produces both the vertical and horizontal sync pulses used in a 525 line per frame system. The method used to generate squares on the CRT is to intensify the successive sweeps identically for one clock period. The largest multiple of 16 (number of rows) that can be used in this system is found to be 30. That is, 30 horizontal lines will be the vertical width of each square. In each interlaced field (one half of a frame), there will be 15 horizontal lines per square. The 30 lines total are sufficient to appear to the eye as a solid square.

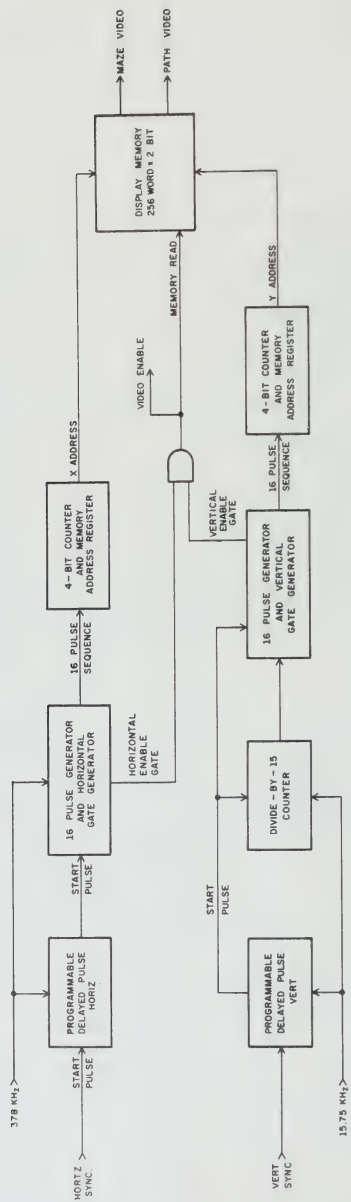


Figure 3. Square Generator Block Diagram

To address the 16 bits per row horizontally a 4-bit binary counter is used. After a selected delay time has elapsed, 16 pulses are generated and used to address each memory location in the row. The 16 pulses counted are derived from the master clock which is at a frequency of 378kHz which is 24 times the horizontal sweep frequency. Therefore there are $24-16=8$ clock intervals which are allotted to blanking and retrace intervals. The first few clock pulses after the horizontal sync are used for a delay count to blank the left side of the CRT display. The right side of the CRT display is blanked by ending the video gate after the sixteenth addressing pulse has occurred.

The operation of the vertical addressing logic is similar to the horizontal logic except that the input is 15.75kHz. It is then divided by 15 to generate the square as explained earlier.

Included in the vertical and horizontal logic are two signals, horizontal and vertical enable gates, which are anded together. This signal is used to gate the memory into the read mode, and signal the video logic generator to receive maze and path information.

Ed Pott

2. HARDWARE SYSTEMS RESEARCH

(Supported in part by the Atomic Energy Commission under Contract
US AEC AT(11-1) 1469, W. J. Poppelbaum, Principal Investigator.)

Summary

Doug Sand's OLFT report describes the temperature sense and control circuits for the Curie temperature crystal arrangement. By means of these, the crystal can be maintained at any temperature down to about -60°C to a tolerance of 0.1°C . Dick Blandford has a block diagram of LINDA and also discusses a video delay and some separation logic for locating parts of a line drawing. Parts of Stereomatrix have undergone extensive redesign. Those described by Shiv Verma are the new coefficient generator, inverse transformer and cursor generator (replete with logic drawings for the unbelievers). Steve Whiteside has also participated in this redesign and outlines a new version of the control box encoder, while shedding a few tears over an acousto-optic deflector and laser power supply. Sik Yuen's Scantrix report deals with the design of various control circuits. Frog's world model is described by Debasish Bose. Finally, Martin Jer discusses a new LED display project called BEACON.

M. Faiman - editor

2.1 LASCOT (Project No. 09)

A Spectra Physics Model 164 mixed gas laser was received. Installation of the laser is in progress.

The mirrors in the scanner were increased in size (Horizontal deflection mirror = 3 x 3 mm, vertical deflection mirror = 1" x 1"). This was done to increase signal to noise ratio at the output of the photomultipliers in the scanner. Experiments are being conducted to determine the effectiveness of this change.

M. N. Cooper

2.2 OLFT (Project No. 12)

2.2.1 Temperature Sense and Control Circuits

The temperature of the crystal and substrate is sensed by four calibrated pn junctions and controlled by a set of thermoelectric modules. The temperature circuits consist of four temperature sensor amplifiers, four temperature difference amplifiers, and five voltage-controlled current drivers. These are discussed below.

The temperature sensors are 2N3128 transistors. Each transistor is contained in a small ceramic-and-epoxy case about 0.05" x 0.05" x 0.03", which allows fast thermal response and requires negligible space when bonded to the substrate near the crystal edge. From the Ebers-Moll equations the voltage V_{BE} is a linear function of temperature at constant I_{BE} and $V_{CB} = 0$. All four junctions were simultaneously calibrated against a thermocouple and two precision thermistors. The temperature range was -60°C to -20°C , with increments of 1°C , and unit-to-unit temperature variation of less than 0.05°C . At $I_{BE} = 10.00\mu\text{A}$, the measured V_{BE} fit the curve $V_{BE} = B - AT$, where $B \sim 570\text{mV}$ and $A \sim 2.4 \text{ mV}/^{\circ}\text{C}$. The sensor circuit in Figure 1 consists of a current

source to bias the junction, and a precision amplifier with adjustable bias and gain to produce $V_{\text{out}} = (T/10)$ volts, where T is the sensor temperature. The overall variation between all four sensors and associated circuits is less than 0.1°C over the calibration temperature range.

Each substrate edge is cooled by a thermoelectric module which is driven by a voltage-to-current power amplifier shown in Figure 3. The voltage input V_{Ii} is determined by the circuit shown in Figure 2. The desired temperature T_0 is selected by setting a precision potentiometer for which voltage follower A_0 generates $V_{T0} = T_0/10$. Emitter follower Q_0 sets a limit for maximum current which is controlled for each module by voltage comparator C_i , with visual indication given by the light-emitting diode MV50. Amplifier A_i generates the current-control voltage V_{Ii} , in the form $V_{\text{Ii}} \approx 22(V_{\text{Ti}} - V_{T0}) - 0.2(V_{T0} + V_{\text{Ti}})$. At the normal operating temperature, about -50°C , the thermoelectric modules require about 2 amps each, and the above equation gives $V_{\text{Ii}} \approx 2$ with $V_{\text{Ti}} = V_{T0}$.

Each of the thermoelectric modules attached to the substrate is cooled by a primary thermoelectric module attached to the copper heat sink. The four primary modules are operated electrically in series, at current $I_5 \approx 6\text{A}$. This current is controlled by the power current source shown in Figure 4, which is essentially a scaled version of the circuit in Figure 3. The unregulated power supplies for all five current sources are indicated in Figure 5.

2.2.2 Photon Couplers

The write-gun circuits require dynamic focussing and improved coupling in the video isolation amplifier, for which several high-voltage photon couplers were constructed by Masamichi Sato. Each unit consists of a Monsanto ME60 infrared-emitting diode as transmitter and a Monsanto

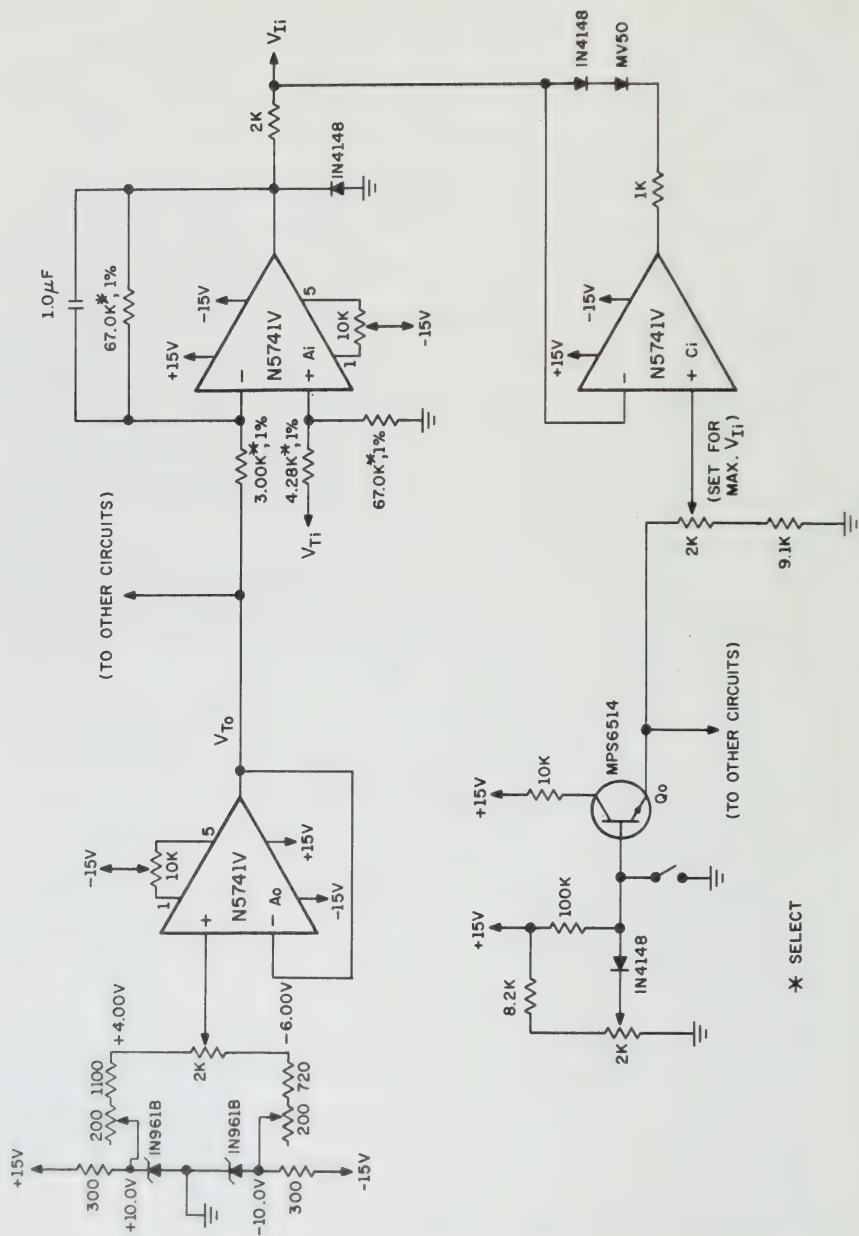


Figure 2. Current-Source Control Circuit

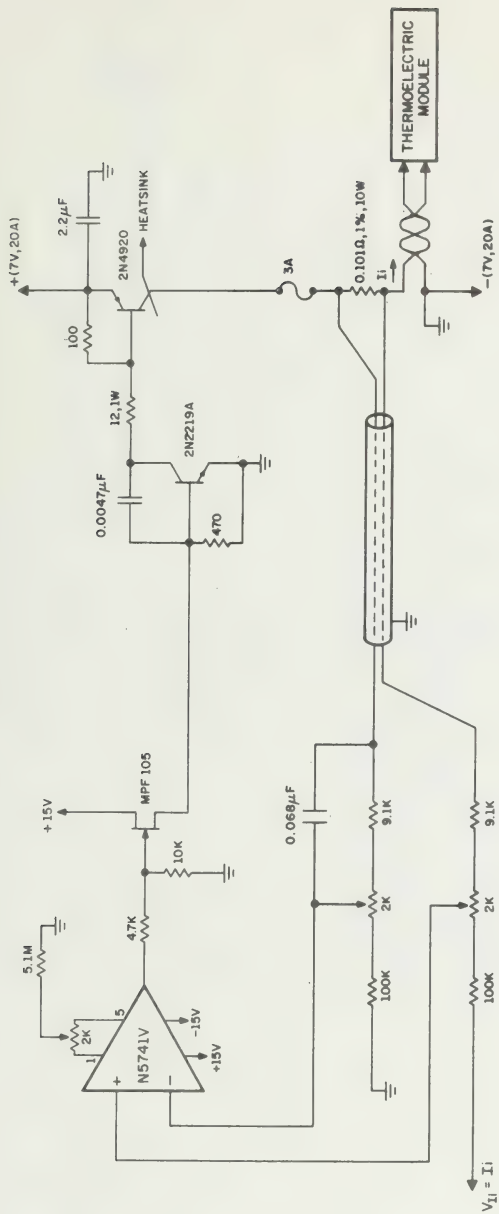


Figure 3. Current Source for Secondary (Colder) Cooling Module

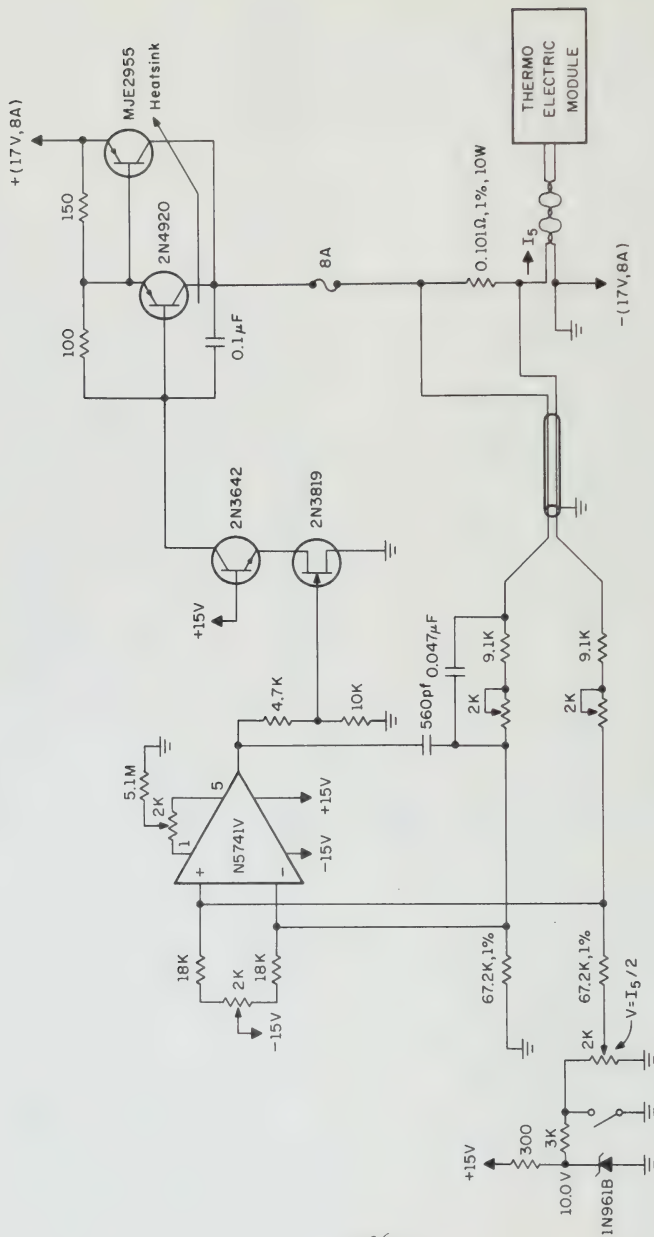


Figure 4. Current Source for Primary (Warmer) Cooling Module

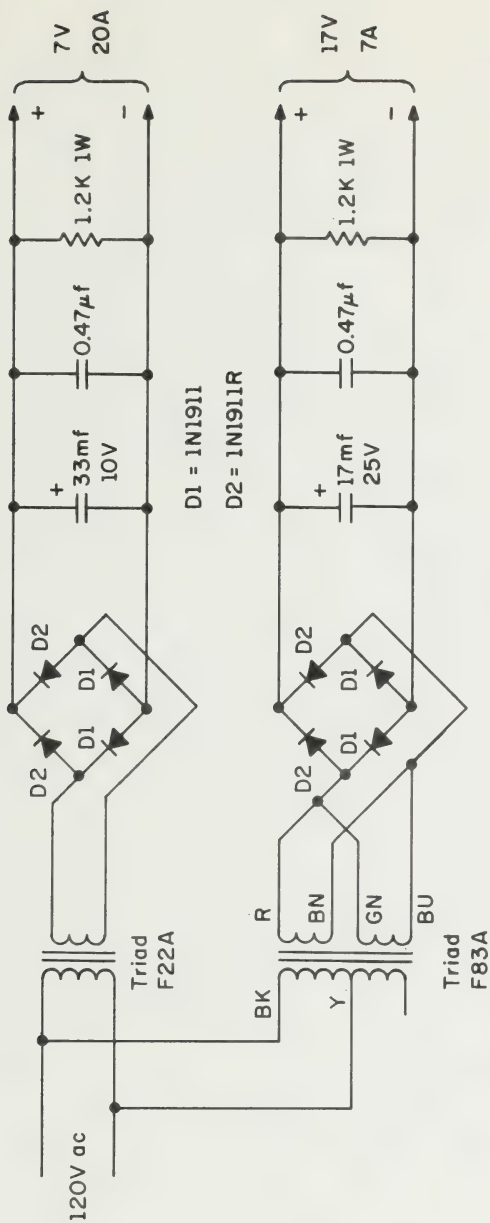


Figure 5. Power Supplies for Cooling-Unit Current Sources

MD2 PIN photodiode as receiver. The diodes are epoxy-bonded to opposite ends of a 3/8" hollow polystyrene tube about 5" long, and are optically connected by an acrylic rod, about 2" long and 1/8" in diameter, which has lens-like ends formed by heating and polishing. Each diode is connected to a Subminax 27-9 jack bonded to the ends of the tube. For continuous currents, the typical transfer coefficient is about 5 μ A out (at 20V reverse bias) for 10mA input. The maximum continuous input current is 40mA. Each diode has a pulse rise-time of less than 1nS, so for well-designed driver and amplifier circuits, the unit rise-time should be less than 5nS. All units were tested to have negligible leakage current at 30kV.

Doug Sand

2.3 LINDA (Project No. 28)

2.3.1 Summary

Project LINDA (Line Drawing Analyzer) is an attempt to build a machine capable of recognizing a small vocabulary of simple line drawings. Figure 1 shows a block diagram of the system. Recognition of a given drawing is made by determining the drawing's simple parts and their relative position. Each drawing is made on a slide used in a flying spot scanner. The video signal from the scanner is analyzed, and erasing and shading operations are performed to divide a drawing into its simple parts. Each part is identified separately by counting discontinuities, and recognition of a given drawing is made when the parts are known, along with their relative positions.

2.3.2 Project Status

The photocell line array, the amplifiers, adder, sweep circuits, and flying spot scanner are in working order from work completed in previous

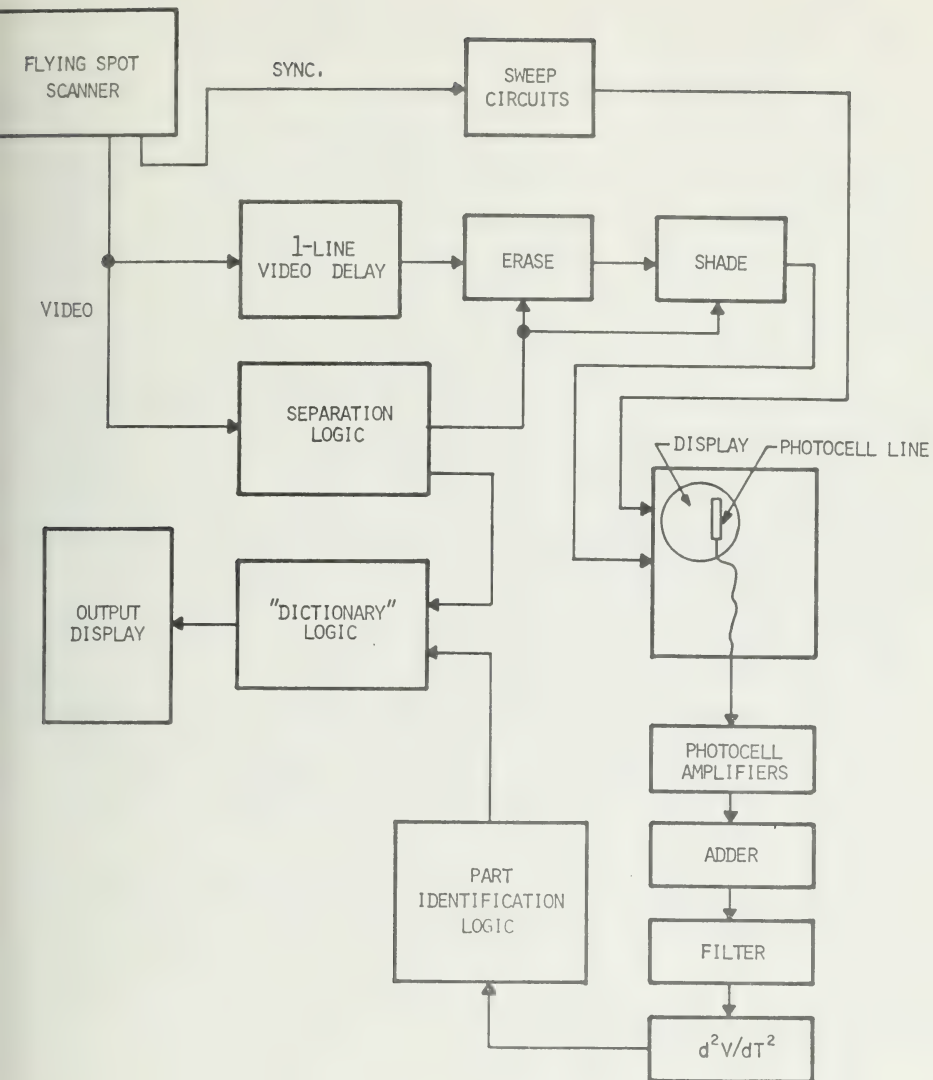


Figure 1.
LINDA block diagram.

quarters. During the past quarter the 1-line video delay was completed and the separation logic was planned, with construction about 60% complete.

2.3.2.1 1-Line Video Delay

The video delay is necessary so that an entire line of video may be looked at before a decision is made as to what action (erase or shading) must be performed on it. The delay is not analog but uses a series of one-shots to delay the rise and fall time of video pulses.

2.3.2.2 Separation Logic

This logic block counts the pulses on a video line and determines where the simple parts of the line drawing are. It erases or shades in each part in turn. Noise problems were encountered in counting pulses. Any miscount of the number of pulses on a video line results in an error in determining where the simple parts of a line drawing are. High contrast line drawings were used to help eliminate this problem and a comparator circuit was built to shape the video signal and help eliminate noise. The comparator circuit consists of two comparators - comparator A samples the video signal at a high level while comparator B samples at a low level, as shown in Figure 2. The output can then go to a high level only if A and B both go high, and the output goes to a low level only if A and B both go low. This circuit has been built and tested.

The line pulse counter has also been completed and cards are being made to compare one line with the previous line to determine changes in the figure. More complete details of this work will be given in future reports when the separation logic is complete.

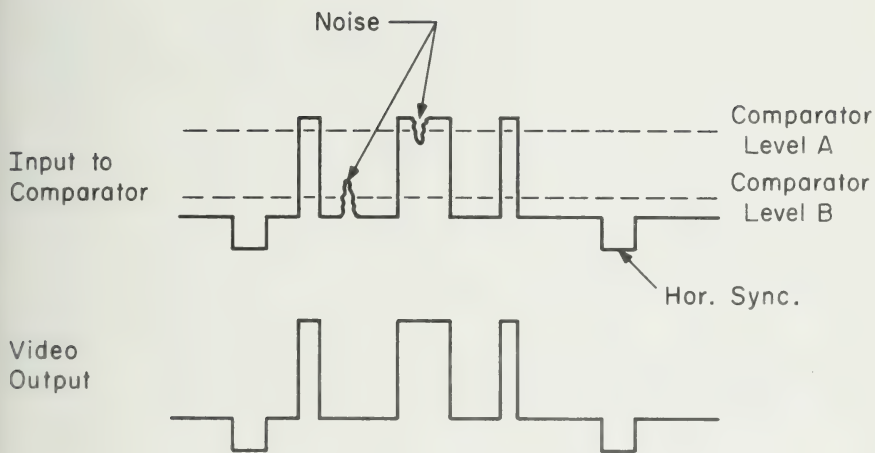


Figure 2.
Typical Video Lines to Illustrate
Function of Comparator Circuit

2.3.3 Future Work

Work during the next quarter will be directed at finishing the separation logic. Hopefully, this can be completed and work can be started on the part identification block and the "dictionary".

Dick Blandford

2.4 RASER (Project No. 29)

In this quarter RASER was almost completed. At the planning stage of RASER we assumed that the input to RASER is 2's complement binary numbers but at this moment we do not have a digital input. We must use an analog input and a couple of 8-bit analog to digital converters to get a digital input to RASER.

By the end of the next quarter RASER will be completed.

Tak Kato

2.5 Stereomatrix (Project No. 30)

2.5.1 Coefficient Generator

This quarter has been devoted to redesigning the Coefficient Generator, a block diagram of which is shown in Figure 1.

Since we are rotating the picture in space by 0.2° per frame, it is necessary to have twenty bits for cosine and sine. Hence, we have two 20 bit by 9 bit multipliers for computing new rotation coefficients from the previous ones. A scheme which uses few packages and has simple control has been used to implement the 20 bit by 9 bit multiplier. Figure 2 shows the logical diagram of the multiplier, its control and timing diagram.

After multiplying previous coefficients with $\sin(0.2^\circ)$ and $\cos(0.2^\circ)$, they are to be added to find new coefficients. Addition is performed on the magnitude and the sign of the result is obtained separately. Figure 3 shows the logical design of the adder, comparator and sign detection scheme.

These two cards have been laid out, wired and are being tested. The control card and control box decoder card have also been laid out and tested. Rack wiring for the Coefficient Generator has been completed. Thus, it is hoped that the Coefficient Generator will be working in a few weeks.

2.5.2 Inverse Transformer and Cursor

The block diagram of the cursor and inverse transformer is shown in Figure 4. The cursor will be displayed as a 3-dimensional cross instead of a circle. Thus, the cursor and cross generator has to be redesigned. Figure 5 shows the diagram of the coincidence detector. This card has been laid out and is being tested. The 3-dimensional cross generator has been designed and its circuit is as shown in Figure 6. The rack wiring for this part is being done.

ROTATION & TRANSLATION COEFFICIENTS

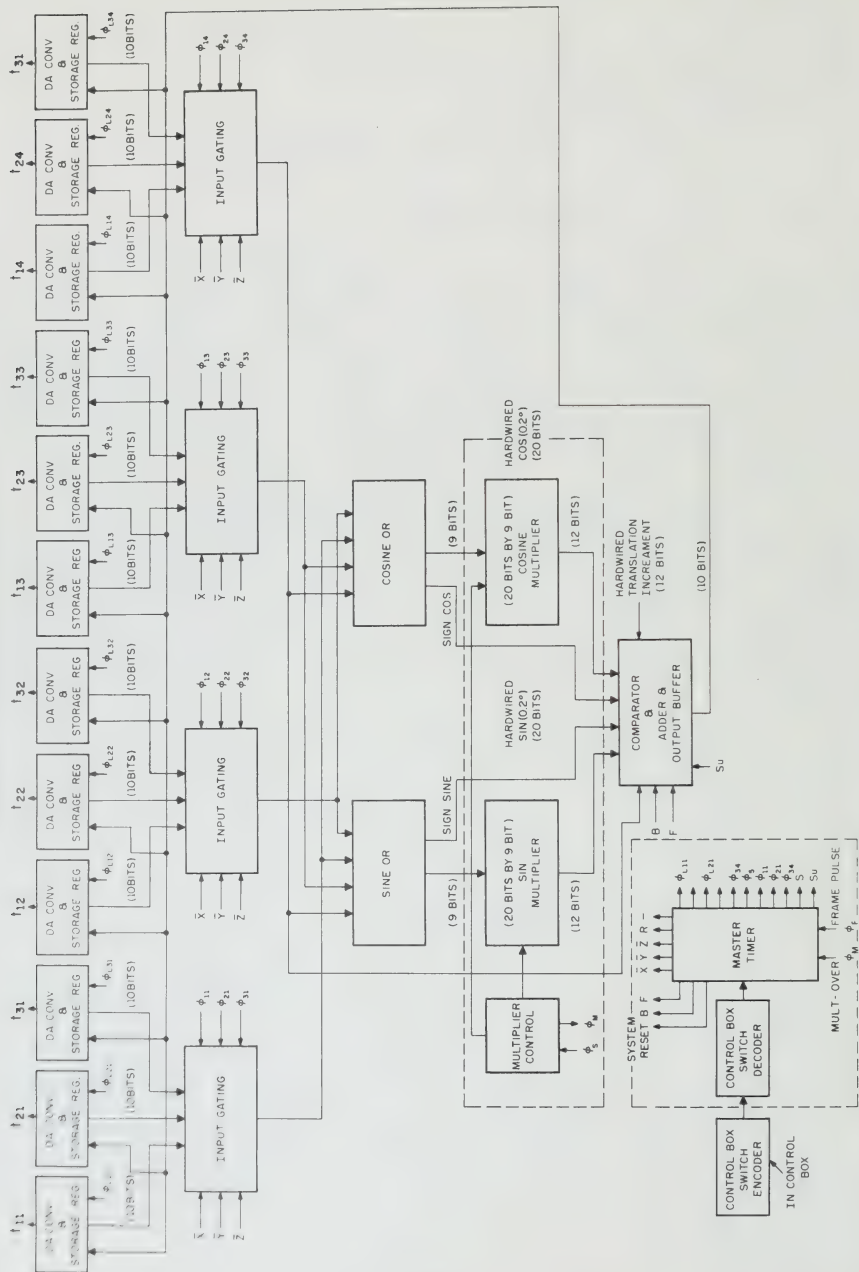
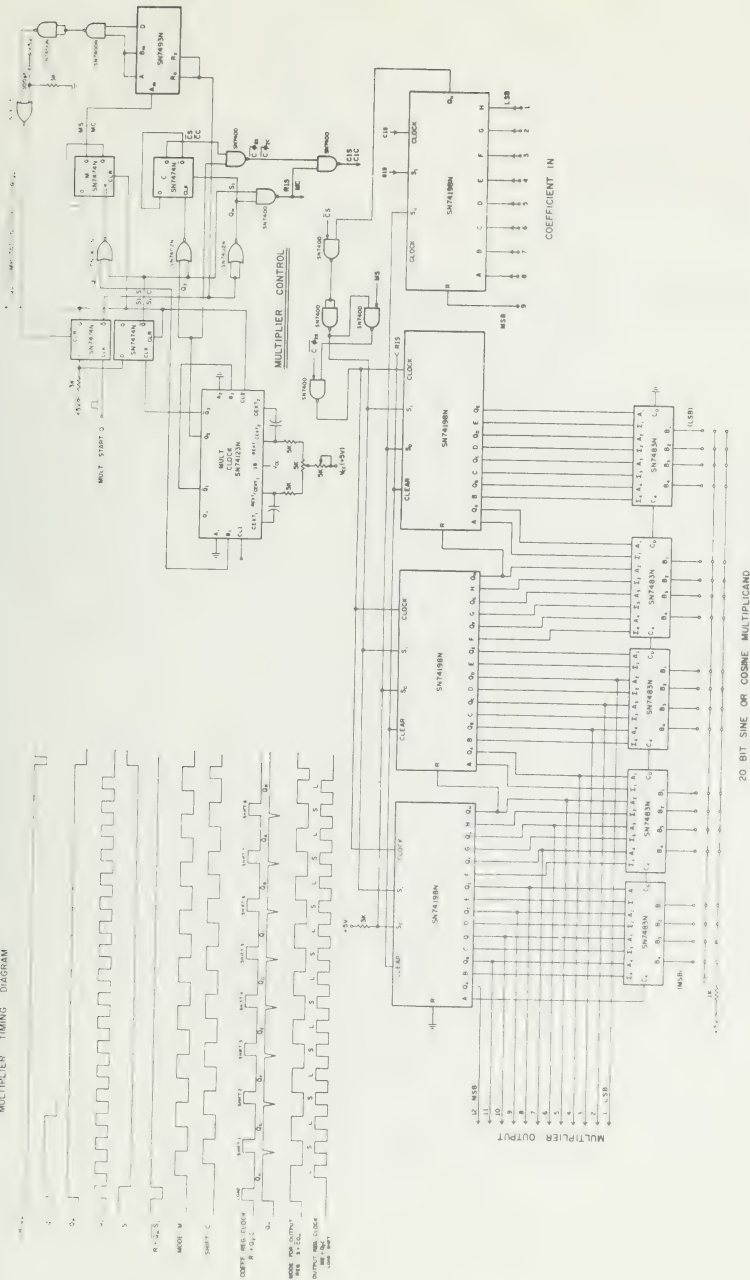


Figure 1. Block Diagram of Rotation and Translation Coefficient Generator



-35-

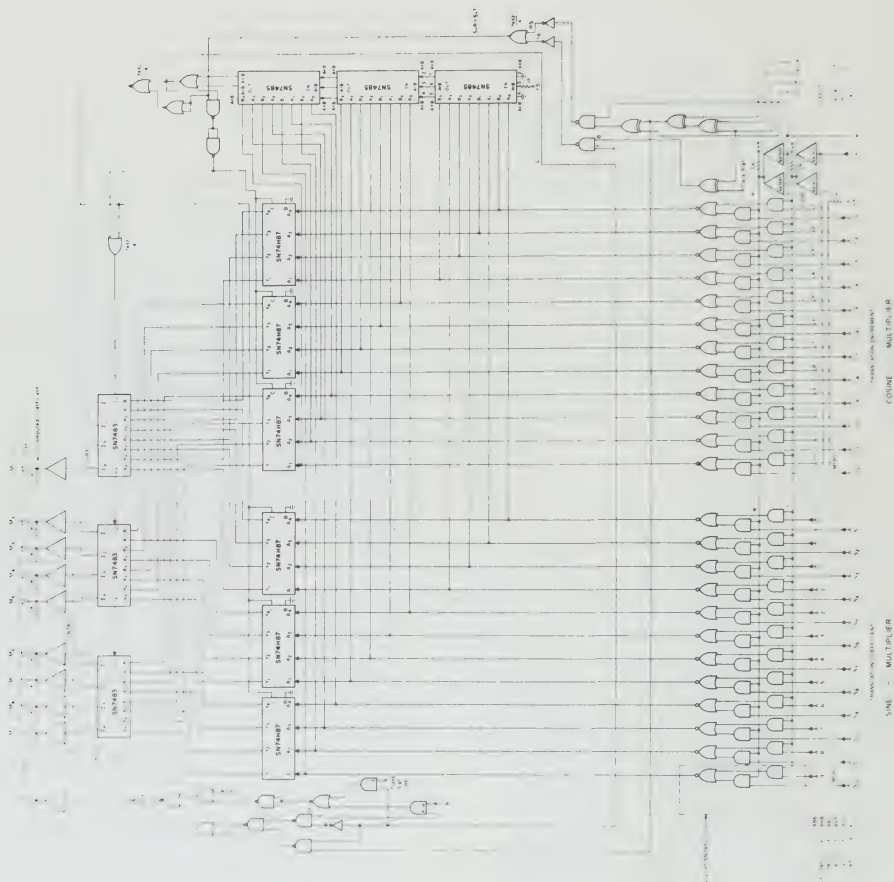


Figure 3. Rotation and Translation Coefficient Computing Card

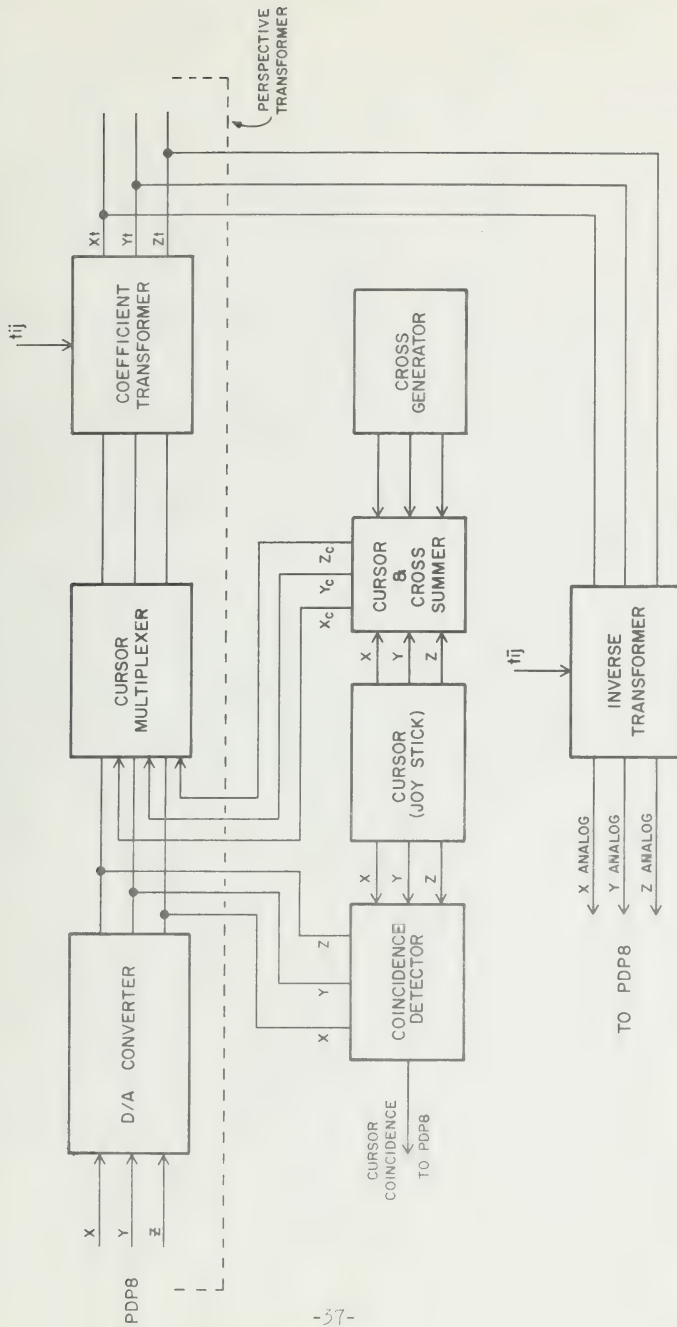


Figure 4. Cursor and Inverse Transformer

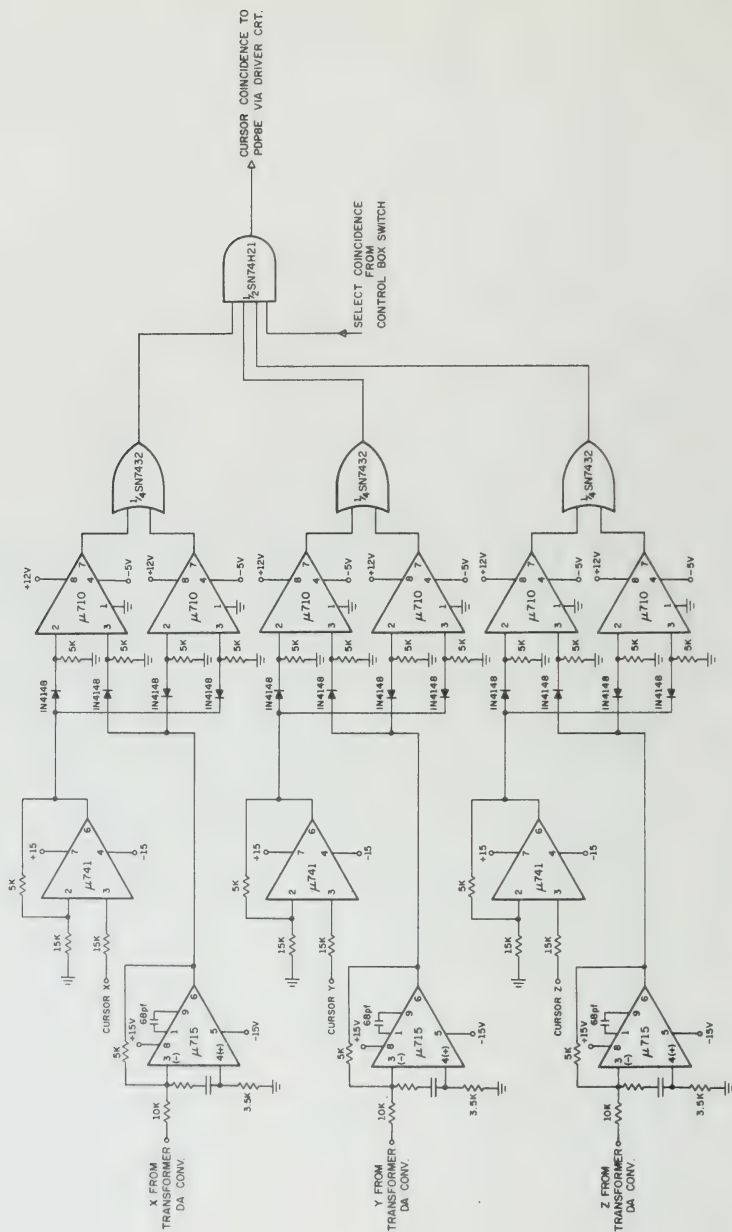


Figure 5. Cursor Coincidence Detector Circuit

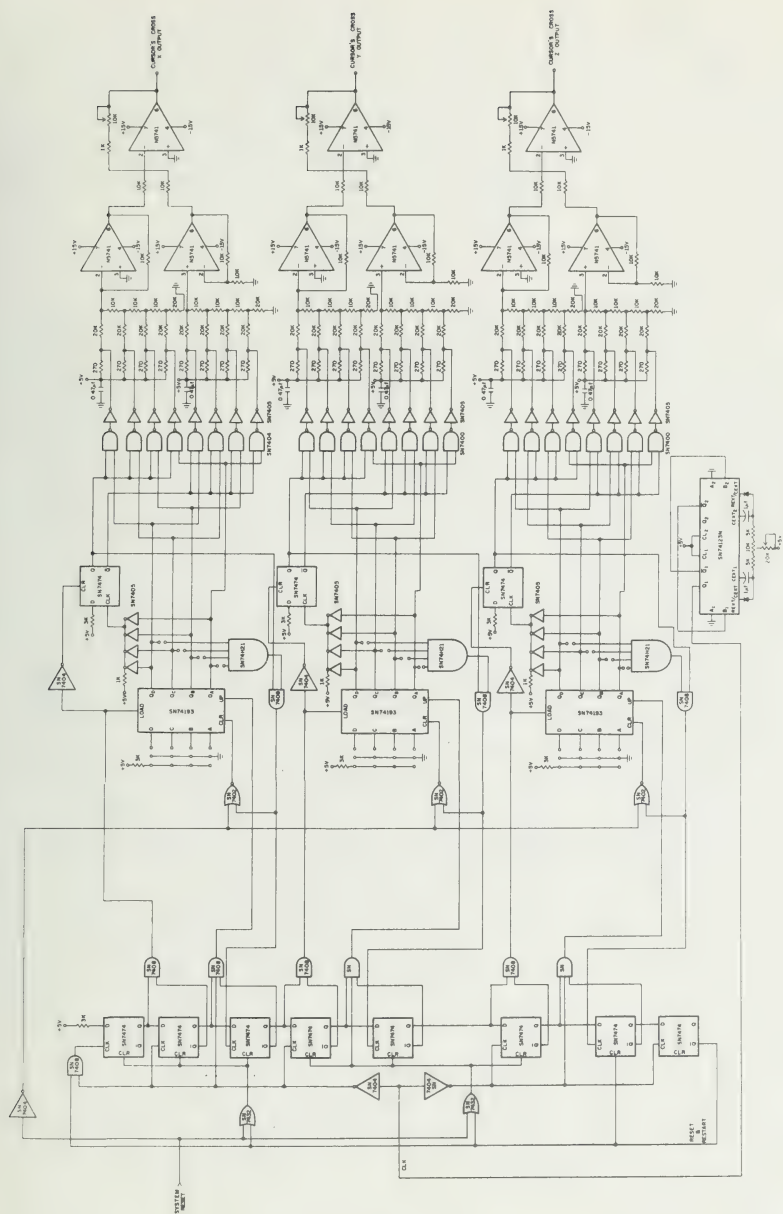


Figure 6. Cursor Cross Generator

Few cards in the inverse transformer have been modified to correct the design. All the cheaper pots are being replaced by good trim pots. The divider and polarity correction circuitry has to be redesigned.

It is expected that part of the inverse transformer will be working at the end of the quarter.

Shiv Verma

2.5.3 Display

Acousto-optic deflector no.3 has been repaired under warranty and returned. This was another electronic driver failure. The unit would follow the manual position control from one side of the screen to approximately center screen and then deflection would stop completely. Returning the position control to the one extreme would regain deflection. These symptoms indicate the the variable capacitance diode at the circuit input was breaking down.

The coated lenses have arrived and will be assembled into a covered box for eye safety and dust protection.

The laser power supply has failed again. Comparing this supply with other manufacturers' is interesting. At 25 degrees C. this current regulator can dissipate 2.5kw. Other units can absorb 5-6kw. At 80 degrees C. dissipation capabilities are 1.16kw and 2.8kw, respectively. Either a complete second current regulator or an automatic voltage control circuit is going to be incorporated in the laser power supply. As it is still possible that the starting circuit transients are causing some of the problems, a stored, fixed amount of energy will be used to start the plasma arc in the laser tube.

2.5.4 Coefficient Generator Redesign

A new control box encoder was designed, constructed and tested. The circuit is given in Figure 7. This is much more efficient than the previous



Figure 8. Stereomatrix Coefficient Register

canonical sum design. The new design includes memory and lamp drivers. Thirteen commands are encoded to 8 wires by essentially 4 separate encoders. Three of the encoders remember and display the last button pushed until a new selection is made. The decoder for these controls is part of the master control card and will be given later.

A new coefficient register was designed, constructed and tested. The circuit is given in Figure 8. This new design eliminates one register. The multiplier input now comes from the output register which also feeds the D/A's. The 51 ohm/220 pF delay holds the reset data at the 7475 inputs until the reset clock input goes away. The 390 ohm/470 pF delay is simply to allow the transfer register to settle before the output register is clocked to read.

Stephen Whiteside

2.6 Scantrix (Project No. 35)

2.6.1 TV Signals

It was decided that a commercial TV would be used as a receiver for TV pictures. The video signals are processed in the TV set as usual. At the appropriate points, video, audio, vertical sync, and horizontal sync signals are taken out, to be further processed in the system. A circuit diagram of the TV set being used is shown in Figure 1.

2.6.2 Composite Video

The composite video can be easily obtained at TP-V (Figure 1) by connecting this point to the input of an LM302 (Figure 2). The LM302 was chosen for its very high input resistance, so that the original video of the TV is not disturbed. The LM302 has an output resistance of 2.5 Ω which is desirable.

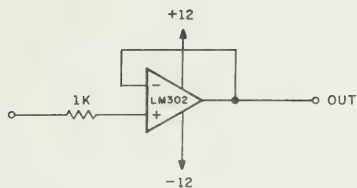


Figure 2. Composite Video

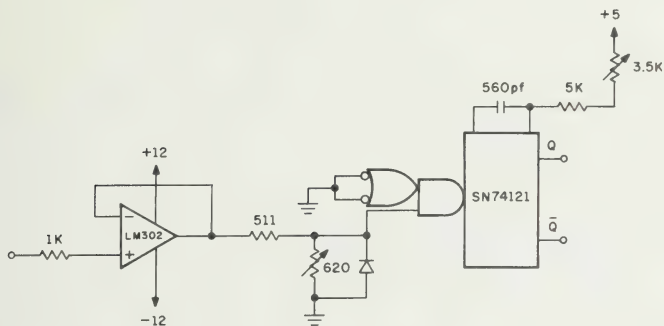


Figure 3. Horizontal Sync

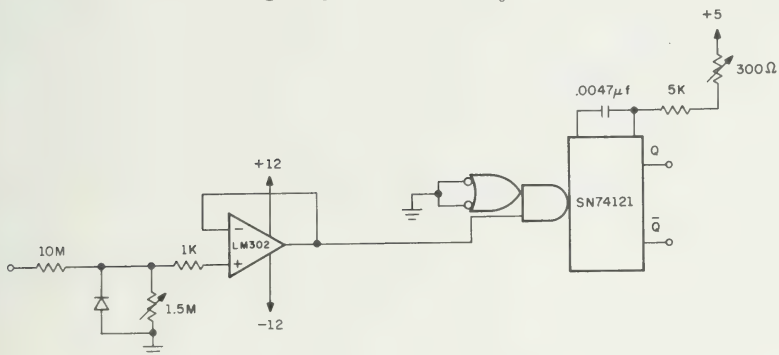


Figure 4. Vertical Sync

2.6.3 Horizontal Sync

A very simple approach was adopted to obtain the sync signals from the TV set. A #30 teflon wire was looped once around point A (Figure 1). Because of the high voltage characteristic (300V p-p) at point A, an output voltage of about 11V p-p can be obtained at the end of the teflon wire. This provides the input for the IM302, the output of which is then attenuated appropriately and subsequently used to trigger the SN74121 whose timing resistors and capacitors are chosen so as to give the correct sync signals for the system.

2.6.4 Vertical Sync

Another teflon wire was used to loop around point B (Figure 1) to obtain the vertical sync signal (600V p-p) which then is processed like the horizontal sync.

2.6.5 Phase-Locked Oscillator (System Clock)

It was described in the last section how the sync signals can be obtained. If the inverse of the horizontal sync is used to trigger an edge-triggered D-type flip-flop (SN7474), the output of the SN7474 will correspond to the rising edges of the input. The output is then used to trigger a pair of cross-coupled one-shots to generate 128 pulses for every interval between each pair of consecutive horizontal sync pulses (Figure 6). Two SN74193's operating in the count-down mode, with an initial value of 128 set every time the sync signal comes along, are used to stop the one-shots every time 128 pulses are counted. It is important to make sure that all these 128 pulses are generated well within the interval between each consecutive pair of horizontal sync pulses in order not to skip any horizontal line of information.

Sik Yuen



Figure 5. Phase-Locked Oscillator

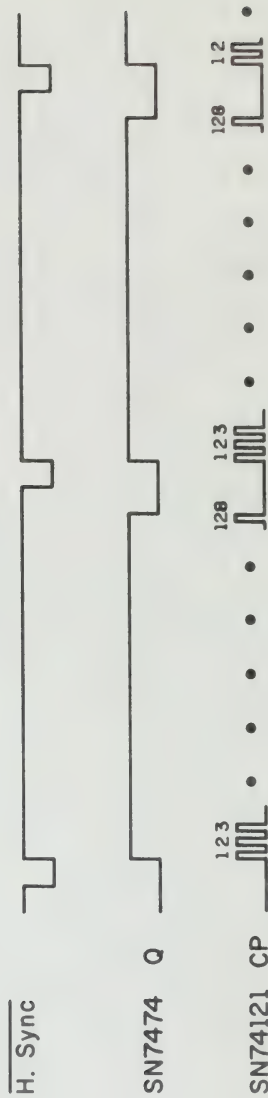


Figure 6. Output of Clock Generator

2.7 Frog (Project No. 36)

2.7.1 Project Status

A PL/1 simulation of the current scheme has been done in this quarter. The results are yet to be analyzed, but it is hoped that the simulation will help in making final design decisions.

2.7.2 The World Model: Structure of the Memory

Some aspects of the structure of the memory were described in the last quarterly report. Figure 1 shows the block diagram of a memory plane. The nonvisual inputs to a memory plane are the good taste (GT) and the bad taste (BT) of the stimuli and the PAIN information. The GTM, the BTM and the PM are each divided into four levels. Level 1 associates the nonvisual input with the four basic visual characteristics of the stimuli. Levels 2, 3 and 4 associate the nonvisual inputs with combinations of the four basic visual characteristics taken two, three and four at a time respectively. The fifteen visual inputs to a memory are

Level 1 visual inputs:

$$(1.1) = \text{visual}_1$$

$$(1.2) = \text{visual}_2$$

$$(1.3) = \text{visual}_3$$

$$(1.4) = \text{visual}_4$$

Level 2 visual inputs:

$$(2.1) = \min (\text{visual}_1, \text{visual}_2)$$

$$(2.2) = \min (\text{visual}_1, \text{visual}_3)$$

$$(2.3) = \min (\text{visual}_1, \text{visual}_4)$$

$$(2.4) = \min (\text{visual}_2, \text{visual}_3)$$

$$(2.5) = \min (\text{visual}_2, \text{visual}_4)$$

$$(2.6) = \min (\text{visual}_3, \text{visual}_4)$$

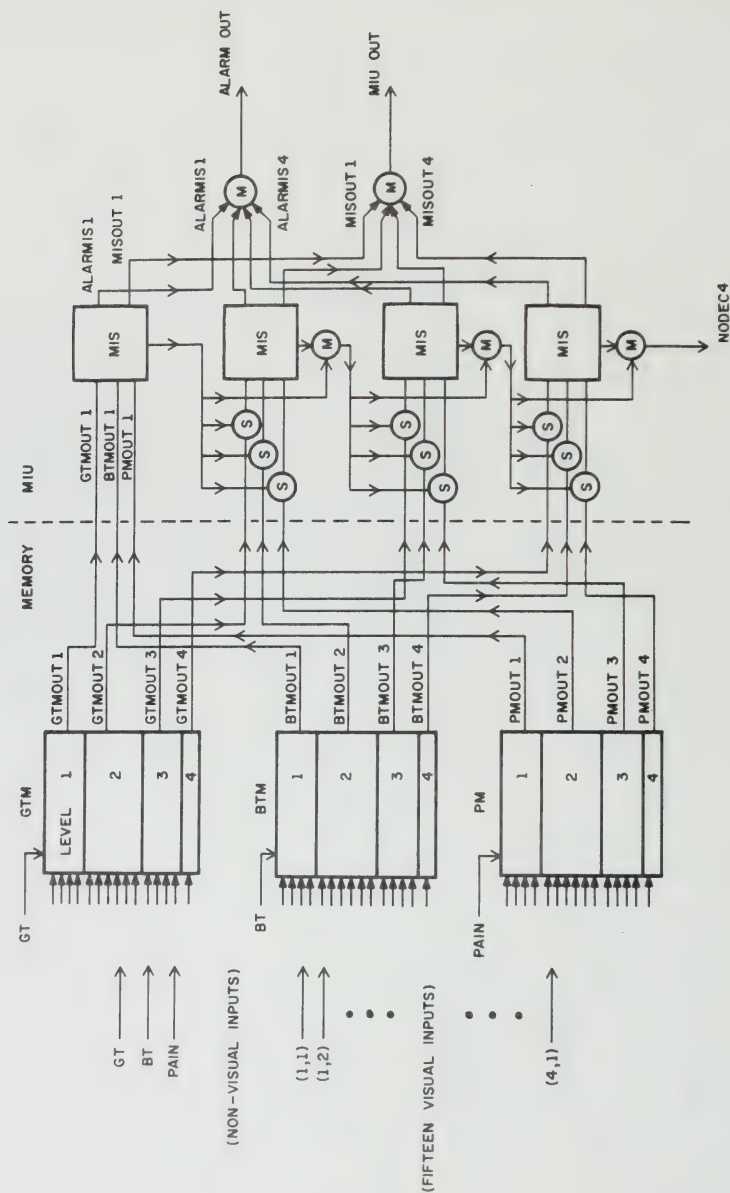


Figure 1. Block Diagram of a Memory Plane

Level 3 visual inputs:

$$(3.1) = \min (\text{visual}_1, \text{visual}_2, \text{visual}_3)$$

$$(3.2) = \min (\text{visual}_1, \text{visual}_2, \text{visual}_4)$$

$$(3.3) = \min (\text{visual}_1, \text{visual}_3, \text{visual}_4)$$

$$(3.4) = \min (\text{visual}_2, \text{visual}_3, \text{visual}_4)$$

Level 4 visual inputs:

$$(4.1) = \min (\text{visual}_1, \text{visual}_2, \text{visual}_3, \text{visual}_4)$$

2.7.3 The World Model: Memory Integration Unit

During a read cycle the visual characteristics are sent to the memory plane corresponding to the region in which the stimulus has been presented. The four level outputs from each of GTM, BTM and PM are then handled by the MIU, one in each plane. The MIU in the plane compares three signals at a time starting with the three level 1 outputs. This comparison is done in an element called the memory integration subunit (MIS), as shown in Figure 1. The outputs from a memory plane are ALARMOUT, MIUOUT and NODECH. If FROG senses a predator it will have an ALARMOUT. If FROG senses a bug there will be an output, MIUOUT, proportional to its desire to eat. But if no decision is reached the NODECH output of the plane becomes 1.0. When this happens FROG gates in the visual inputs to all the other planes for a readout from them. This enables FROG to come to a decision based on its knowledge of this stimulus in all the regions. The maximum of the MIUOUT of all the planes is the final MIUOUT (Figure 2). The ALARMOUT outputs of all the planes are also passed through an M-element for the final ALARM output. If no decision is reached in all the planes the NODEC will become 1.0. Under this circumstance MIUOUT is equal to 1.0 if HUNGER is greater than 0.5 and 0.0 if HUNGER is less than or equal to 0.5.

Debasish Bose

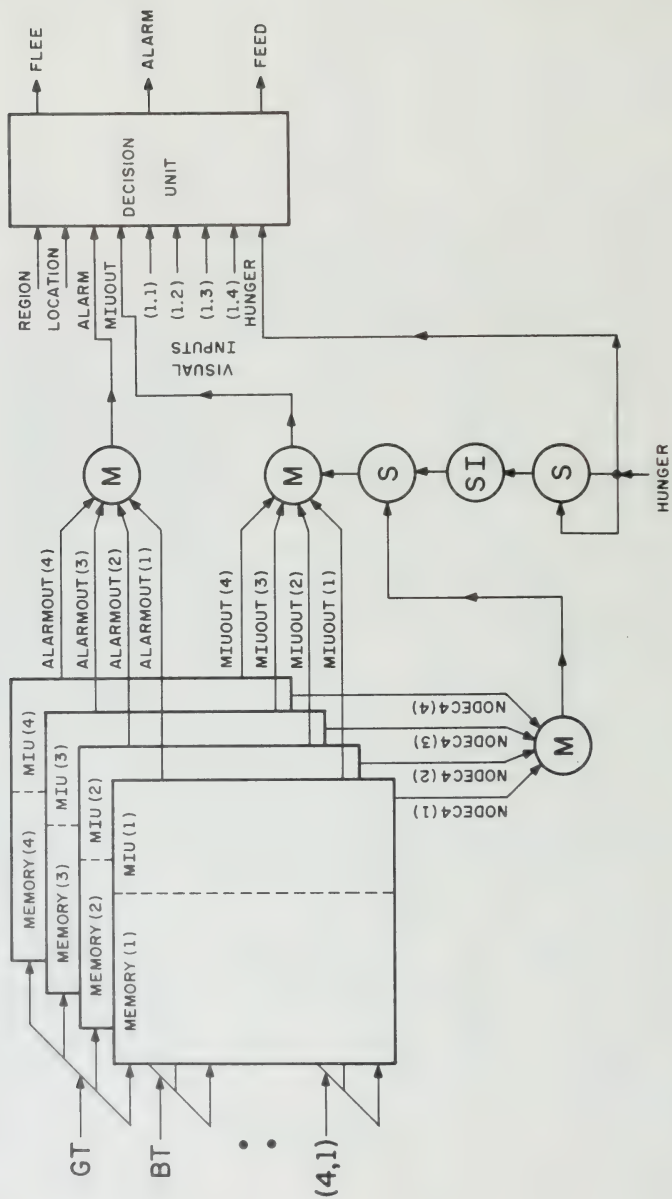


Figure 2. Block Diagram of the World Model

2.8 BEACON (Project No. 38)

2.8.1 Introduction

One of the objects of using LEDs in displaying a TV picture is to make it stay on for one picture time (that is about 30 ms) in order to get a bright picture. The purpose of the Beam Control project is to utilize the phosphor decay time on a CRT as a memory to keep the picture information. Fiber optics or light guides are used to transmit light from the phosphor screen to a light detector. (Figure 1) The signal from the light detector is amplified to drive the LEDs. The amplifier is also equipped with a switch so that a particular picture can be frozen by activating the switch.

2.8.2 Work Done

The main difficulty now is to find a proper Fiber Optic faceplate. CRT price quotes from the manufacturers are outrageously high.

2.8.3 Alternative Method

In a regular television picture tube, the phosphor layer is 1/4 to 1 inch away from the outside surface. The photo transistor to be used here has a lens built on it to allow incoming light to focus on the transistor chip. The transistor is mounted on a board with holes arranged in a square matrix (Figure 2a). The thickness of the board can control the viewing area (Figure 2b). This arrangement allows each transistor to "see" its own area on the phosphor screen, but not the area of the neighboring transistor (Figure 3).

2.8.4 Future Work

Work will be done in designing the most economical circuit for the amplifier.

Martin Jer

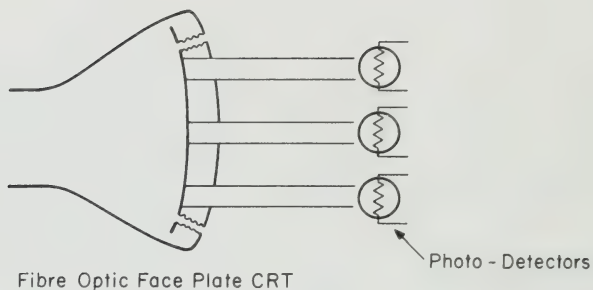
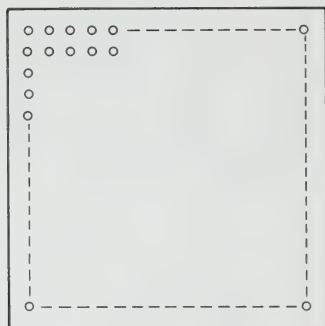


Figure 1.
Fiber Optic Faceplate CRT



32 x 32

Figure 2a.
Phototransistor Board

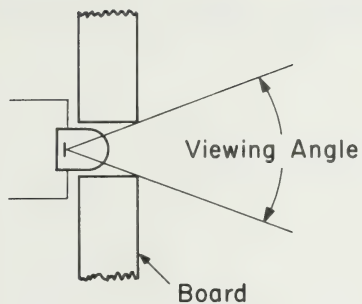


Figure 2b.
Viewing Angle

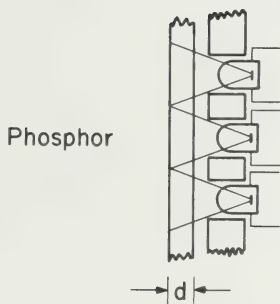


Figure 3.
Faceplate Mounting

PUBLICATIONS

Charles R. Pirnat, "Observer Position Detector for the Stereomatrix 3-D Display System", Department of Computer Science Report No. 491, University of Illinois, Urbana-Champaign, February 1972.

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

C00-1469-0206

2. TITLE

Quarterly Report - January, March, February

3. TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

- ☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

W. J. Poppelbaum
Principal Investigator

M. Faiman, Editor
Associate Professor of Computer Science

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois

Signature

W. J. Poppelbaum

Date

May, 1972

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

3.1 Numerical Processes3.1.1 Ordinary Differential Equations (R. L. Brown)

DIFSUB received one new improvement, one significant change, and previous improvements were incorporated into the program.

The major improvement involved a look-ahead feature in the corrector iteration convergence test common to both Adams method and the stiff method. It follows from the observation that the L_2 norm of the i -th corrector term, $||e_i||_2$, is approximately $R*||C_{i-1}||_2$ where R is approximately constant throughout a small region of integration and is initially computed from $R = ||C_2||_2/||C_1||_2$. Later values are found from the maximum of $.9*R$ (previous R) and $||C_2||_2/||C_1||_2$ at each step. Since the L_2 norm is required to compute R , the corrector convergence test was changed from the max norm to an L_2 norm test

$$\text{MIN}(|C_i|, R*|C_{i+1}|*2) < \text{BND}.$$

The factor of two insures that the next corrector term will be less than $1/2$ of BND and, therefore, need not be computed.

Since three corrector terms is optimal for most cases, and the look-ahead feature approximates the third corrector from the second; only two corrector terms are now found.

Since inverting the PW matrix for stiff methods is inefficient compared to Gaussian elimination and back substitution for five or more variables, DIFSUB was changed to use a GE and a back substitution set of subroutines.

Also, the Adams method was brought up to 12-th order, test having shown that many problems would use orders higher than even, and this leads to improved convergence.

Finally, a new step size is used for only eight steps, not ten, before testing for a further change in step size.

This version was run with Krogh's test and found to produce 10% to 30% faster convergence as measured by the number of times the differentiating subroutine was called.

3.1.2 Sparse Matrix Inversion (J. S. Deogun)

The sparse eigen problem is closely related to the sparse matrix inversion in that the same set of functional routines (e.g. pivot selection, elimination, etc.) is employed with changed values of some parameters. The eigen value program is nearing completion. The program has been successfully run on small test examples as well as large practical examples. Two major factors need consideration in the program.

First, there is no simple way of checking the accuracy of the results obtained. In fact, because of the nature of the problem, it is not possible to develop a CHECK program as was done for SPARSE. Presently, effort is, therefore, directed towards developing a reasonably good technique of checking the accuracy of the results obtained. One suggested method consists of generating DIFFUN within the program from coefficient matrices (A and B), so that variations of a SYSTEM $Ax = \lambda Bx$ with known eigen values may be tried.

Secondly, some difficulty arises after the original eigen problem has been reduced to a small non-sparse eigen problem. Sparse routines are strong enough to handle numbers over a wide range. In the course of the reduction process in many examples, some very small and/or very large numbers are introduced. These numbers are essentially correct but cause trouble (DIVIDE CHECK, OVER_FLOW, etc.) while inverting the matrix B by

standard (non-sparse) techniques. However, this problem is inherent only in some examples and many examples behave very nicely.

The program was also tested with two different pivot selection strategies. First, the usual one which selects pivots from matrix A or B--whichever is under consideration without seeing the other. In the second strategy, selection of a pivot is based upon the total number of elements in corresponding rows (or columns) of both matrices. Contrary to expectations, the two strategies did not show any considerable difference in the cost--measured as a function of the number of calls made to COMPIL.

3.1.3 The Steady-State Package (B. van Melle)

DIFMF3 is a subroutine used to solve the steady-state problem of ordinary differential equations. It will operate on a mixture of differential and algebraic equations, and can be used merely to solve a system of simultaneous equations, which it handles in the same manner. To solve the system, a combination of the Newton method and a first-order predictor-corrector method is used.

The Method

The problem is presented to us in the form

$$f(y, y', t) = 0.$$

However, for the steady-state problem, t is constant, and in each component of the vectors y and y' either the y or the y' is held constant, so the problem can be written

$$f(y) = 0 \tag{1}$$

where the vector y contains the unknowns.

To solve (1) we invent an independent variable s , and attempt to integrate

$$\frac{df}{ds} = -f \quad (2)$$

from 0 to ∞ . The exact solution of (2) is

$$f = f(y_0)e^{-s}$$

which approximates $f = 0$ as s becomes large. Hopefully, a reasonable solution is obtained while s is still finite.

To obtain an expression for $y' = dy/ds$, apply the chain rule to (2)

$$\begin{aligned} \frac{\partial f}{\partial y} \cdot \frac{dy}{ds} &= -f(y) \\ y' &= -\left(\frac{\partial f}{\partial y}\right)^{-1} f(y) \end{aligned} \quad (3)$$

We use Euler's method for a predictor:

$$y_n = y_{n-1} + hy'_{n-1} \quad (h \text{ the step size}) \quad (4)$$

For a corrector we use the formula:

$$y_n = y_{n-1} + h(\alpha y'_n + (1-\alpha) y'_{n-1}) \quad (5)$$

with $0 \leq \alpha \leq 1$. When $\alpha = 0$ the corrector has no effect, and when $\alpha = 1$ and $h = 1$, the entire method is simply Newton's method

$$y_n = y_{n-1} - \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_{n-1}).$$

Rewriting (5) and substituting for y' ,

$$y_n - h\alpha y'_n = y_{n-1} + h(1-\alpha) y'_{n-1}$$

$$y_n + h\alpha \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n) = y_{n-1} + hy'_{n-1} - \alpha hy'_{n-1}$$

Now write $y_n = y_n^{(0)} + \Delta y_n$, where $y_n^{(0)}$ is the predicted value from (4)

and Δy_n is what we seek an expression for.

$$y_n^{(0)} + \Delta y_n + h\alpha \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)} + \Delta y_n) = y_n^{(0)} - \alpha hy'_{n-1}$$

Use a first-order Taylor expansion to approximate $f(y_n^{(0)} + \Delta y_n)$:

$$\Delta y_n + h\alpha \left(\frac{\partial f}{\partial y}\right)^{-1} [f(y_n^{(0)}) + \left(\frac{\partial f}{\partial y}\right) \Delta y_n] = -\alpha h y'_{n-1}$$

$$(1+h\alpha) \Delta y_n = -h\alpha \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)}) - \alpha h y'_{n-1}$$

$$\Delta y_n = -\frac{\alpha}{1+h\alpha} [h \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)}) + h y'_{n-1}] \quad (5)$$

To obtain a similar expression for $h y'_n$, approximate (3) with a first-order Taylor expansion and substitute from (5)

$$\begin{aligned} h y'_n &= -h \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n) \\ &= -h \left(\frac{\partial f}{\partial y}\right)^{-1} (f(y_n^{(0)}) + \left(\frac{\partial f}{\partial y}\right) \Delta y_n) \\ &= -h \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)}) - h \left[-\frac{\alpha}{1+h\alpha} [h \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)}) + h y'_{n-1}] \right] \\ &= -\frac{1}{1+\alpha h} [h \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)})] + \left(\frac{\alpha h}{1+h\alpha}\right) h y'_{n-1} \\ &= h y'_{n-1} - \frac{1}{1+\alpha h} [h \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)}) - h y'_{n-1}] \end{aligned}$$

Thus, the corrector step can be written

$$\Delta = \frac{1}{1+h\alpha} [h \left(\frac{\partial f}{\partial y}\right)^{-1} f(y_n^{(0)}) + h y'_{n-1}]$$

$$y_n = y_n^{(0)} - \alpha \Delta$$

$$h y'_n = h y'_{n-1} - \Delta$$

By varying α between 0 and 1 we can alternate between the Newton method (if h is near 1) and this predictor-corrector method, hopefully choosing the more effective scheme for the situation. The Newton method is quite efficient when in the neighborhood of a solution, but can be erratic elsewhere, while the other method generally finds a solution, but very slowly.

DIFMF3

A. Use

The user supplies several subroutines to be used by DIFMF3. The system to be solved is defined by the routines S1, S2, and DIFFUN. S1 and S2 perform computations involving only time and global variables, which remain constant during solution of this problem (these routines may be dummy). DIFFUN consists of equations of the form $DY(J) = J\text{th equation}$. The vector DY then holds the values of the functions $f(y, y', t)$ described in the previous section. The variables y are separated into linear variables which appear without derivatives, stored in the vector YL, and variables with derivatives, which are stored in the first row of an array Y (in $Y(1,*)$) and their first derivatives with respect to time stored in the second row ($Y(2,*)$).

A subroutine MATSET computes the Jacobian matrix $(\frac{\partial f}{\partial y})$, and routine MATINV computes its inverse. MATMUL performs the multiplication $(\frac{\partial f}{\partial y})^{-1} f(y)$. The present DIFMF3 uses routines which utilize sparse techniques, but ordinary dense matrix routines could also be used.

If any of the equations in DIFFUN inherently places a restriction on the range of values y can assume, e.g. a square-root function, another routine, RANGER, must be supplied. This routine checks the restricted variables. If they are out of bounds, they are adjusted to legal values and a return flag is set to reflect this.

In the simplest form of the problem, the derivatives in $Y(2,*)$ are all set to 0, and some sort of initial guess is given to the variables in $Y(1,*)$ and $YL(*)$. The user may wish instead to give a $Y(1,J)$ a constant value and solve for its derivative $Y(2,J)$. The information about which variables to solve for he supplies in the vector IND, whose J-th component is 1 or 2, if he wants to solve for $Y(1,J)$ or $Y(2,J)$, respectively.

B. Strategy

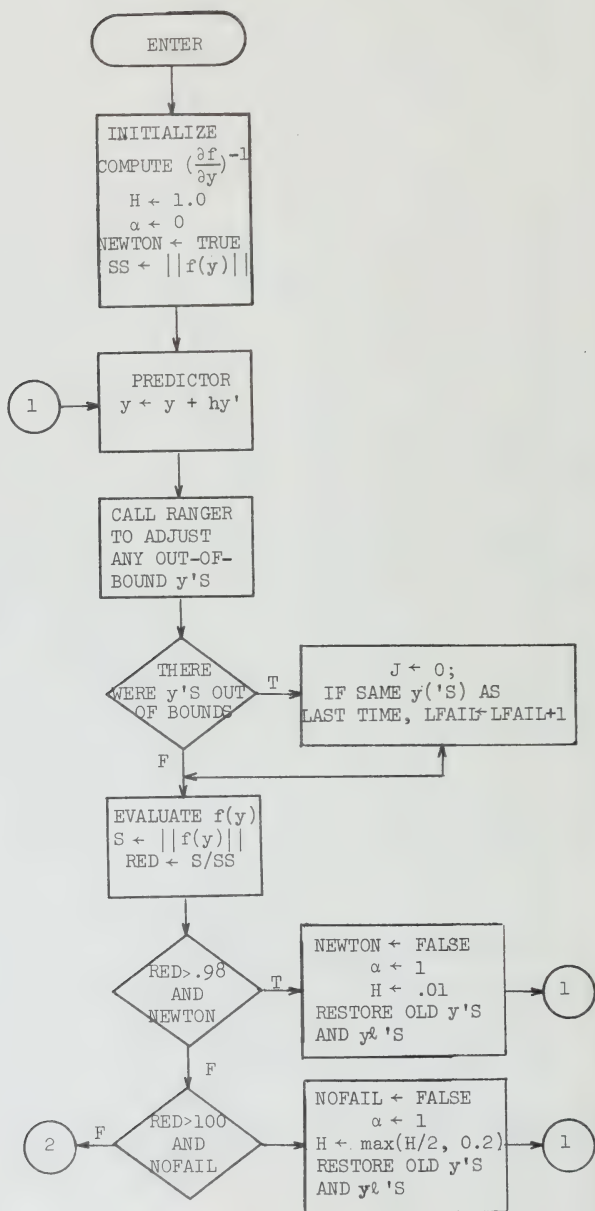
There are two major items of concern in writing DIFMF3:

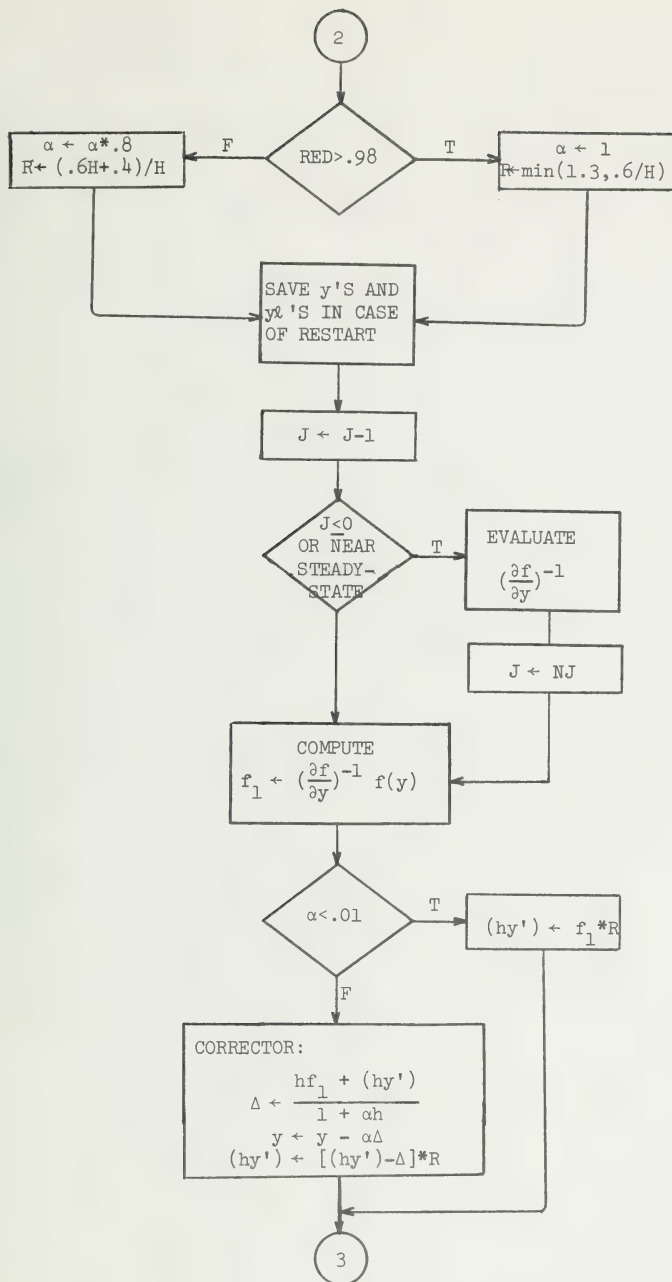
- (1) how to vary α and h to achieve a solution as efficiently as possible,
- (2) how often to evaluate the Jacobian $\partial f / \partial y$.

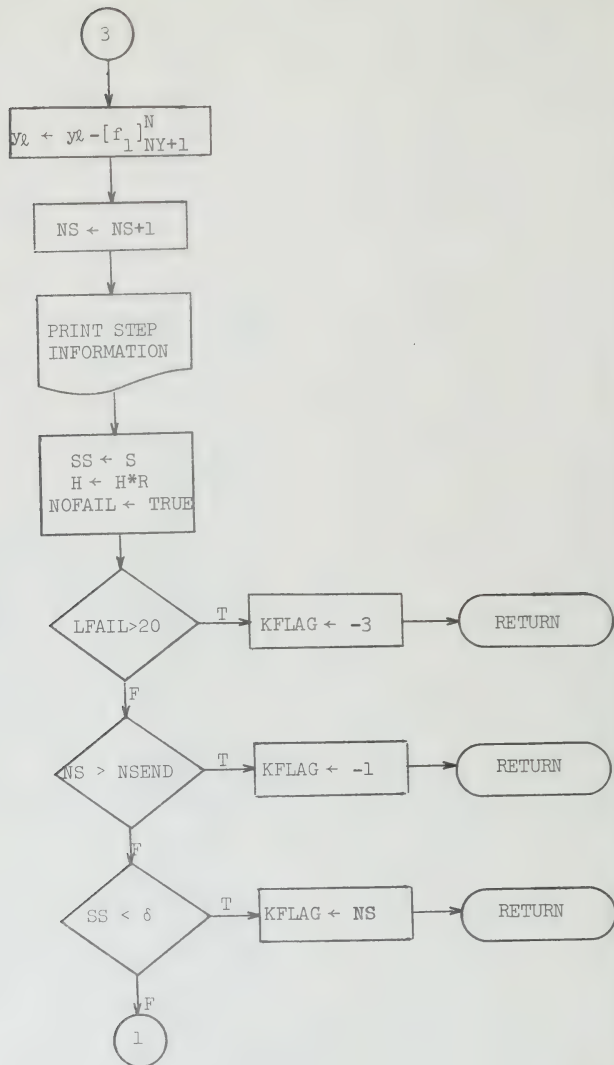
1. We have seen that $\alpha = 0$, $h = 1$, corresponds to the Newton method, while $\alpha = 1$ is the more cautious integration procedure. The former method is quite effective for systems which are linear or nearly so, and also for most systems when near a solution; however, it is potentially quite erratic in other situations, where the latter method must be used. In order to cater to those systems best solved by Newton, we start out immediately with $\alpha = 0$, $h = 1$, and monitor convergence. We continue as long as the "error" $||f(y)||$ decreases. If at any point the error increases, we switch over to the other method, setting $\alpha = 1$ and h small, and continue to monitor convergence. At each step, h is slightly increased. If the error starts reducing favorably, we increase h more rapidly and start phasing out the corrector step by reducing α . If the error later increases, we revert to the $\alpha = 1$ and small h method. The rate used to vary α and h was empirically determined.

2. Computing and inverting the Jacobian is a fairly time-consuming process, so it is desirable to avoid it whenever possible. Fortunately, the Jacobian does not usually change too rapidly, so we can take several steps with the same one. The present strategy calls for recomputing the Jacobian every $NY*5$ steps (NY the number of columns in Y) unless the situation requires otherwise. Such situations include: a large increase in $||f(y)||$, variables going out of bounds, and apparently imminent approach of a solution. The recomputation occurs in the last case because a "fresh" Jacobian speeds convergence considerably when near a solution.

C. Flowchart







D. Explanation of Symbols

Symbol used in flowchart and/or previous discussion)	Symbol used in DIFMF3	Explanation
α	ALPHA	See part II
Δ	D	See part II
δ	DEL	Convergence criterion: equations are considered satisfied when $S < DEL$
$f(y)$	DY	Vector containing the values of the functions $f(y)$
	EPS	Passed to MATSET
	EQN	Passed to MATSET
f_1	F1	Vector containing the product $(\partial f / \partial y)^{-1} f(y)$
	G	An array of global variables
h	H	The step size
	HINV	Held constant at 1.0 for this program
	HOLD	Value of H associated with the saved values of $Y(3,*)$
	IND	A vector indicating which of the Y variables to solve for
J	JACOB	A flag indicating when to re-evaluate the Jacobian. Decrement at each step, when it becomes zero or negative the Jacobian is computed and $JACOB \leftarrow NJ$.
MATINV	MATIN1 MATIN2 MATIN3	Subroutines which divide up the labor of computing the inverse Jacobian.
KFLAG	KFLAG	A return code (see program listing)

LFAIL	LFAIL	Number of times a given set of variables is found out of bounds. When excessive causes termination of DIFMF
	LFLAG	Return code from RANGER. Bits set correspond to variables out of bounds.
	LFLAG1	Previous non-zero value of LFLAG. It is AND'ed with LFLAG to see if the same variables are guilty as before.
	LIST	Describes bounds on Y variables; used by RANGER.
	M	Number of equations in DIFF
	MF	Method indicator (see program listing).
	MM	Dimension of LIST, PLIM; used by RANGER
	N	NL + NY = total number of variables. Usually M = 1
NEWTON	NEWTON	A flag indicating whether we are still performing the NEWTON method which started the routine
NJ	NJ	The Jacobian is evaluated every NJ steps, barring trouble.
	NJJ	NJ-5. A comparison of NJ and NJJ is made to avoid too frequent evaluation of the Jacobian even in trouble situations.
	NL	Number of variables in YL.
NOFAIL	NOFAIL	Flag used to avoid a continuous loop when S suddenly increases.
NS	NS	The step number, incremented at each step.
NSEND	NSEND	The maximum number of steps to attempt before termination

	NW	Number of calls to MATSET.
	NY	Number of columns in Y.
	PLIM	Contains bounds used by RANGER.
red	PRED	S/SS. An indication of the rate of convergence.
	PW	Contains the inverse Jacobian.
R	R	The factor by which H is changed at each step; used to scale Y(3,*).
S	S	$ f(y) = \sum_{i=1}^M DY(i) ,$ a measure of the distance from the solution.
SS	SS	Value of S from previous step.
	SAVE	An array used to save the variables Y(IND(J), J), Y(3,J) in case of mishap.
t	T	Time and remainder variables, constants to DIFMF3.
	VAR	Used by MATSET.
y,hy'	Y	An array containing in Y(1,*) dependent variables Y(2,*) their derivatives with respect to t Y(3,*) hy' Y(IND(*),*) corresponds to y.
yl	YL	An array of variables appearing only linearly and without derivatives.
	YSLV	A vector used to save the YL variables.

E. Examples

Some of the systems used to test DIFMF3 follow. Pseudo-random starting values were used, $t = 0$, and the convergence criterion was $\delta = 10^{-6}$.

Most of these systems are purely algebraic. The last three include restricted variables. Solutions to these often take longer because variables go out of range and are shoved around a lot by RANGER before they settle down. System (3) is linear, so the solution was obtained quite swiftly.

$$1. \quad 4 + y_1 + y_2 - y_1^2 + 2y_1y_2 + 3y_2^2$$

$$1 + 2y_1 - 3y_2 + y_1^2 + y_1y_2 - 2y_2^2$$

$$\text{Starting values:} \quad y_1 = -2.057 \quad y_2 = -7.503$$

$$\text{Solution obtained:} \quad y_1 = 3.339 \quad y_2 = -2.984$$

$$NS = 24, NW = 5$$

$$2. \quad y_1^2 + y_2^2 + y_3^2 - 5$$

$$y_1 + y_2 - 1$$

$$y_1 + y_3 - 3$$

$$\text{Starting values:} \quad y_1 = -2.057 \quad y_2 = -7.503 \quad y_3 = -4.834$$

$$\text{Solution obtained:} \quad y_1 = 1.667 \quad y_2 = -.6667 \quad y_3 = 1.333$$

$$NS = 28, NW = 4$$

$$3. \quad t_2 + y^x_1 - 6y_3 - y'_3$$

$$y_3 - y^1_2$$

$$y_2 - y^1_1$$

$$y^x_3 + 11y_2$$

$$y^x_2 + 6y_1$$

$$y^x_1 - y^x_2 - y^x_3$$

$$t_2 = \sin(t)$$

$$\text{Starting values: } y_1 = -2.057, y_2 = -7.503, y_3 = -4.834$$

$$y^x_1 = 4.473, y^x_2 = -6.240, y^x_3 = 9.308$$

$$\text{Solution obtained: } y_1 = y_2 = y_3 = y^x_1 = y^x_2 = y^x_3 = 0.000$$

$$NS = 2, NW = 2$$

$$4. \quad \frac{1}{2} \sqrt{4 - y_1^2} + y_2 - 1$$

$$2y_1^3 + \ln(y_2 + .8) - .136$$

$$\text{Restriction: } -2 \leq y_1 \leq 2, y_2 > -.8$$

$$\text{Starting values: } y_1 = -.9433, y_2 = 3.951$$

$$\text{Solution obtained: } y_1 = 5.394, y_2 = .03705$$

$$NS = 27, NW = 5$$

$$5. \quad \tan(y_1) + y_2^3 - 3y_{1,2} - .5$$

$$\sin(2y_1) - 1/y_2 + 2y_{1,2} - 1$$

$$y_2 + y_{1,2} - 1.5$$

$$\text{Restriction:} \quad \pi/2 < y_1 < \pi/2, y_2 > 0$$

$$\text{Starting values:} \quad y_1 = -.2983, y_2 = 4.751, y_{1,2} = -4.834$$

$$\text{Solution obtained:} \quad y_1 = .7854, y_2 = 1.000, y_{1,2} = .5000$$

$$NS = 116, NW = 14$$

$$6. \quad y_1^2 + y_2^2 + y_{1,2} - 3$$

$$y_2^2 + y_3^2 + y_{2,3} - 10$$

$$\sqrt{2-y_1} + y_{1,2}$$

$$y_{1,2} + y_{2,3} - 3$$

$$20y_3 - 9y_{2,3} + 5$$

$$\text{Restriction:} \quad y_1 \leq 2$$

$$\text{Starting values:} \quad y_1 = -2.057, y_2 = -7.503, y_3 = -4.834,$$

$$y_{1,2} = 4.473, y_{2,3} = -6.240$$

$$\text{Solution obtained:} \quad y_1 = -2.000, y_2 = -1.000, y_3 = 2.000,$$

$$y_{1,2} = -2.000, y_{2,3} = 5.000$$

$$NS = 20, NW = 3$$

3.1.4 Plot Package (W. Chung)

Several attempts were made to improve the flexibility and convenience for user interaction and reduce the program size and running time. Three subroutines were written to support the plot package with these goals in mind.

1. LABEL (IPL0T,IPY,IPYL,FLABEL)

A short assembler language program which produces the labels (character string output) for the curves given the various names as input. When IPY Y-variable and IPYL YL-variable names are given in IPL0T(*), then the character string labels are put in the array FLABEL(*) .

This subroutine needs to be called only when there is overall change in the set of variables to be plotted--deletion of variables can be done by just changing FLABEL(*), for example. Thereby, the needless effort of calling TEXT several times each time the new coordinate of the plotting points was removed.

2. SUBROUTINE SPLINE (X,Y,NPNT,C,S,XINT,YINT,ITAG)

A Fortran subroutine for curve fitting using the cubic spline technique. This can be used for generating more points on the screen to produce smoother curves or for computing the function values which were not provided explicitly.

The basic idea of cubic spline is to try to connect two points of the graph $Y(X)$ without distorting the overall shape of the curve. Given the values $X(I)$ and $Y(X(I))$, $I = 1, 2, \dots, N$, $I = 1, 2, \dots, N$, values of X which lie between $X(I)$ and $X(I+1)$, the value of

$$Y(X) = C(1,I)*(X(I+1)-X)**3+C(2,I)*(X-X(I))**3+C(3,I)*(X(I+1)-X)+C(4,I)*(X-X(I)).$$

The coefficient $C(J,I)$ are computed using the information at all points ($I = 1, 2, \dots, N$).

Several test examples which have been run indicate that the method works well for X near the middle of X(1) and X(N) but gives rather discouraging error for X close to the end points.

Inverse interpolation gave the same result. This will lead to further study and improvements.

Now the package is logically divided into two modes--plot and command modes. Plot mode is entered when the plot package is brought into core or may be re-entered by specifying "OLOTMODE" in command mode. In this mode, user-typed input is accepted and the data structure is constructed to give graphical output. Command mode is entered automatically following the graphical display to provide the user with listed commands and variable names. The user can choose any of the commands by joystick. Variable names can be specified by pointing to the curve or the variable name from the menu area. The program determines the input by analyzing the coordinates or sequencing. To find the variable name which is specified by pointing to the curve, I is determined such that specified X lies between TSV(I) and TSV(I+1) by approximating the first guess

$$K_0 = \frac{(X-120)}{800} * IPNT + 1$$

and linear search, and then the variable name is determined by linear interpolation.

Listing of Commands

- | | |
|--------------|---|
| 1. RESTART | the whole system is restarted |
| 2. DELETE | any of plotted variables can be deleted |
| 3. SHIFT | the coordinate axis can be shifted up and down |
| 4. LIST | saved values of any variable is listed on the screen |
| 5. EVALUATE | function values at specified point is computed by SPLINE (for both Y and T) |
| 6. REFER | a kind of user manual displayed on the screen |
| 7. SAVE | only one specified curve is saved with all others erased |
| 8. ERASE | the screen is cleared |
| 9. DISPLAY | present data blocks are displayed |
| 10. PLOTMODE | plotmode is entered |
| 11. QUIT | good-bye to the plot package |
| 12. RECOVER | for recovery from any user error at any time |

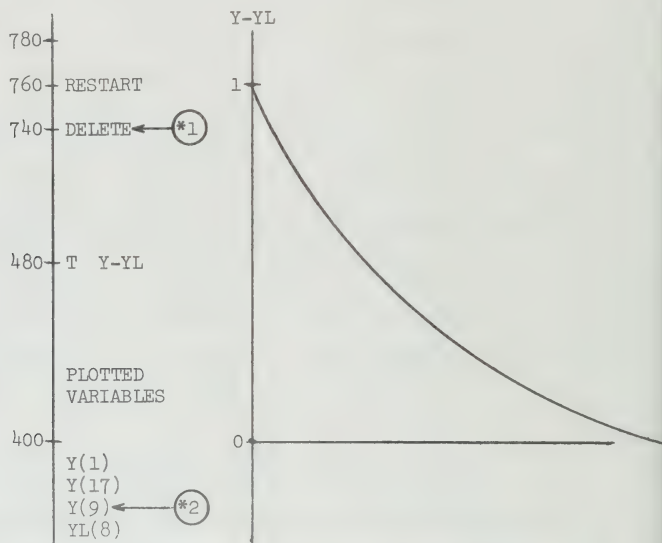
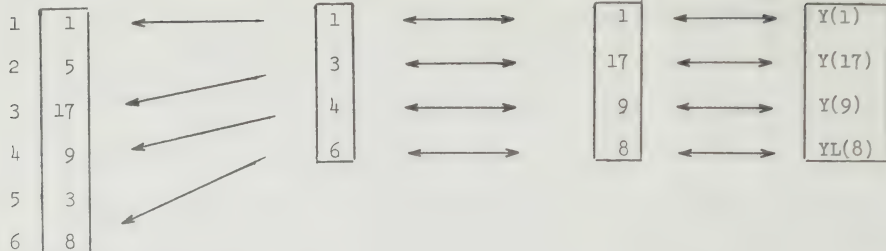
One example will be given to illustrate the simple manipulation of data structure for changing the picture and the screen layout in command mode. In this example, the values of Y(1), Y(5), Y(17), Y(9), YL(3), and YL(8) were saved in YSV(*) and the curves of Y(1), Y(17), Y(9), and YL(3) are displayed.

ISAVE(15)

INDEX(15)

IPLOT(15)

FLABEL(15)



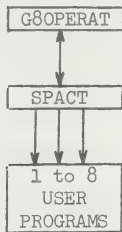
Successive inputs (denoted by *1 and *2) are received to delete Y(9) from the screen, simple recognition of input and easy change of data structure is facilitated by an array INDEX(15) which contains index pointer to the saved variables. INDEX, INPLOT, FLABEL are changed by squeezing action in one DØ loop and the resulting picture generation is no problem.

3.1.5 Matrix Inversion Package (A. Whaley)

3.1.5.1 Modifications to SPACT

Input/output handling in SPACT was changed this quarter.

Formerly, the program arrangement was as follows:

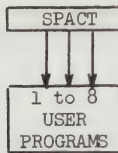


in which SPACT performed the following functions:

1. start, terminate, and intercept abends for user programs
2. handle I/O between user programs and the user
3. handle serialization of user I/O request given to G8OPERAT
4. process user commands used to help user in connection with his user program
 - a. start (S)
 - b. request halt (halt)
 - c. force halt (zap)
 - d. log on (user name)
 - e. log off (logout)
 - f. get core dump (dump)

and G8OPERAT actually performed user I/O, routed messages between SPACT and the 360 operator, and permitted restarting SPACT if the latter should abnormally terminate.

Currently, the program arrangement is as follows



SPACT performs the user I/O itself and communicates with the operator. The restart function has been eliminated in favor of a reliability approach.

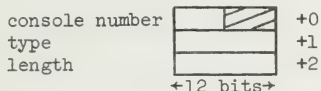
Operator commands now are the same as user commands (except that they do not have to be preceded by !). Commands that deal directly with a user program are assumed to be for console number zero when they come from the operator. In addition to the six commands listed previously

>NN MESSAGE	Send message to console number NN.
>OP MESSAGE	Send message to operator.
STAY	Prevent SPACT from terminating due to excessive time of inactivity. Re-informs PDP-8 periodically of "UP" status.
NOSTAY	Disables STAY.
ISNAP/NIOSNAP	For dumping records transmitted to and from the PDP-8 for which no I/O errors occur.
BOMB	Abnormally terminate SPACT, providing a dump.
GOAWAY	Terminate SPACT, but do not send "DOWN" message to PDP-8.
GRASSOUT	Terminate SPACT and so inform PDP-8.

Considerable code was added to allow the 360 to simulate (very crudely) the ailing datadisk. As our new disk has now arrived, it is not anticipated that this code will be needed again (hopefully), so it will not be described.

The first three words in each record to and from the PDP-8

contain the following information:



Each PDP-8 word occupies two bytes in the 360, six bits per byte. The console number is in the first half of the first word. The remaining first word is not used. The first length is given in the number of PDP-8 words minus one. of data. The data immediately follows these three words and is the only other information transferred in the record. If the total record length is over 256 bytes (the length word would contain over 124), then the record is sent in two parts. The first part is precisely 256 bytes long, and the second part is precisely the length of the remaining data. The length is not normally considered to be a signed number, but a length with all bits set is taken to be -1, or a null data field.

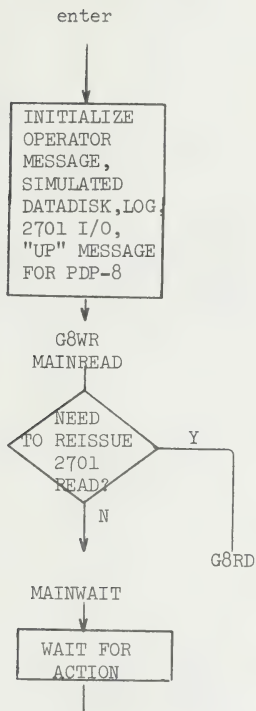
A log of data transmission errors and timeouts is currently kept by SPACT. It is printed on ddname SPACTPRT together with messages to the operator. When a transmission error occurs, the IOB and data buffer are printed on the log. The command IOSNAP causes normal records to be printed as well. All information is printed in hexadecimal as well as normal character format.

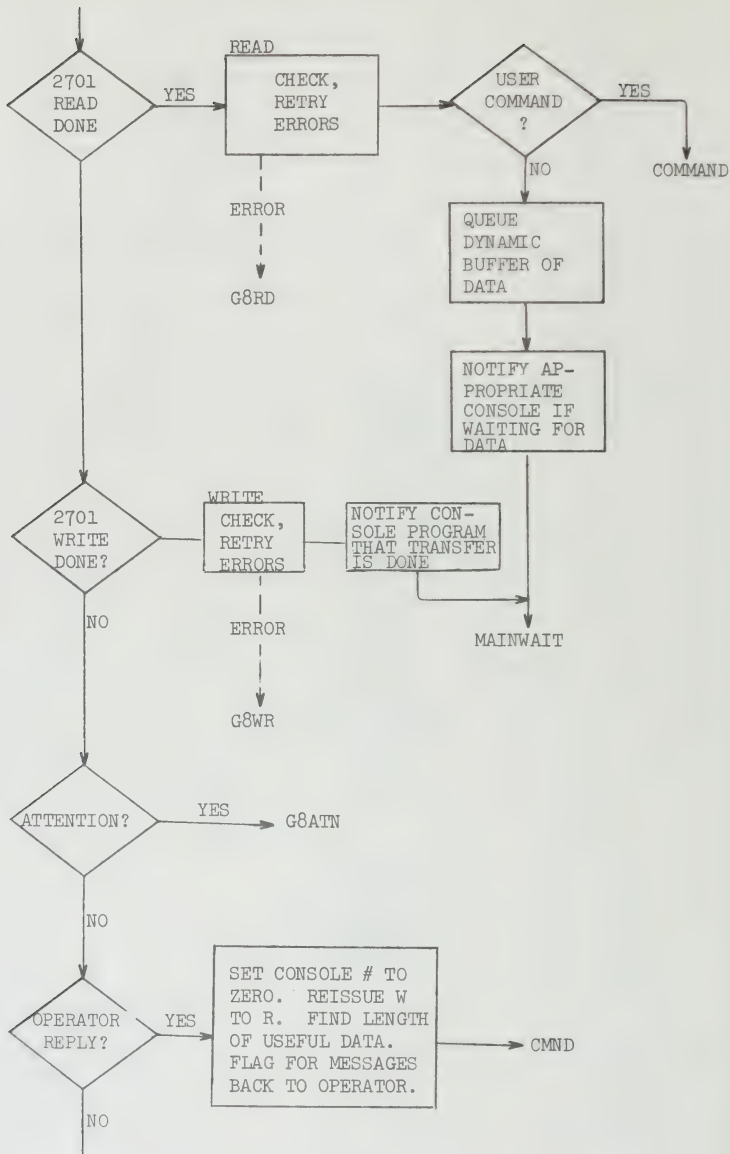
Currently, all records to and from the PDP-8 are queued in SPACT by placing them on a linked list. A 32 bit word immediately preceding the three PDP-8 control words described above is used for this purpose. The user program, however, uses this word to indicate his total buffer length as getmained (getmain means to obtain core), and SPACT currently incorrectly regards this as the transmission length, which is

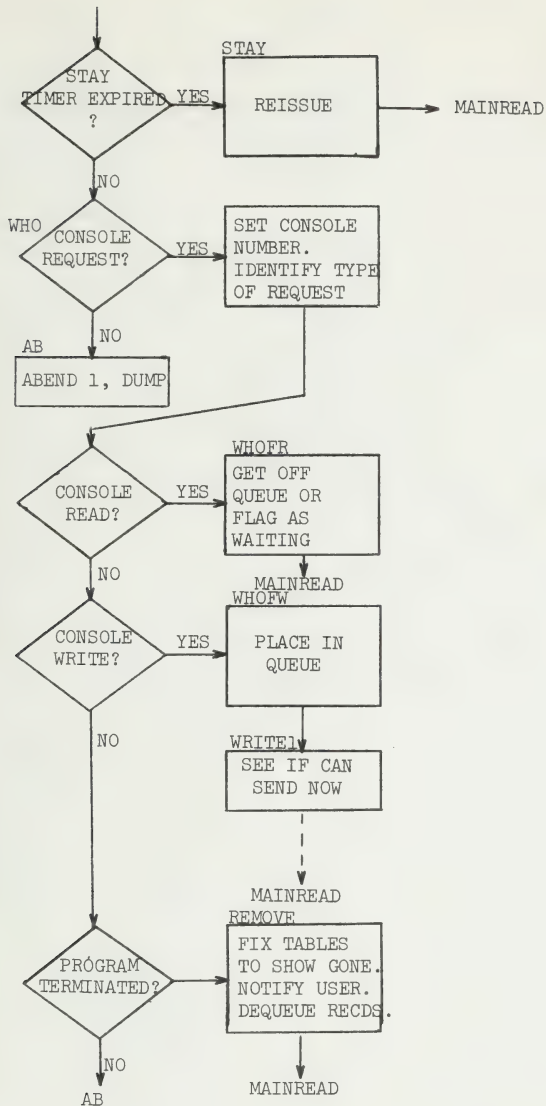
actually four bytes less as the initial 32 bit length word is not transmitted. This scheme is unsatisfactory, however, as all of an oversize buffer will be transmitted rather than just the useful data.

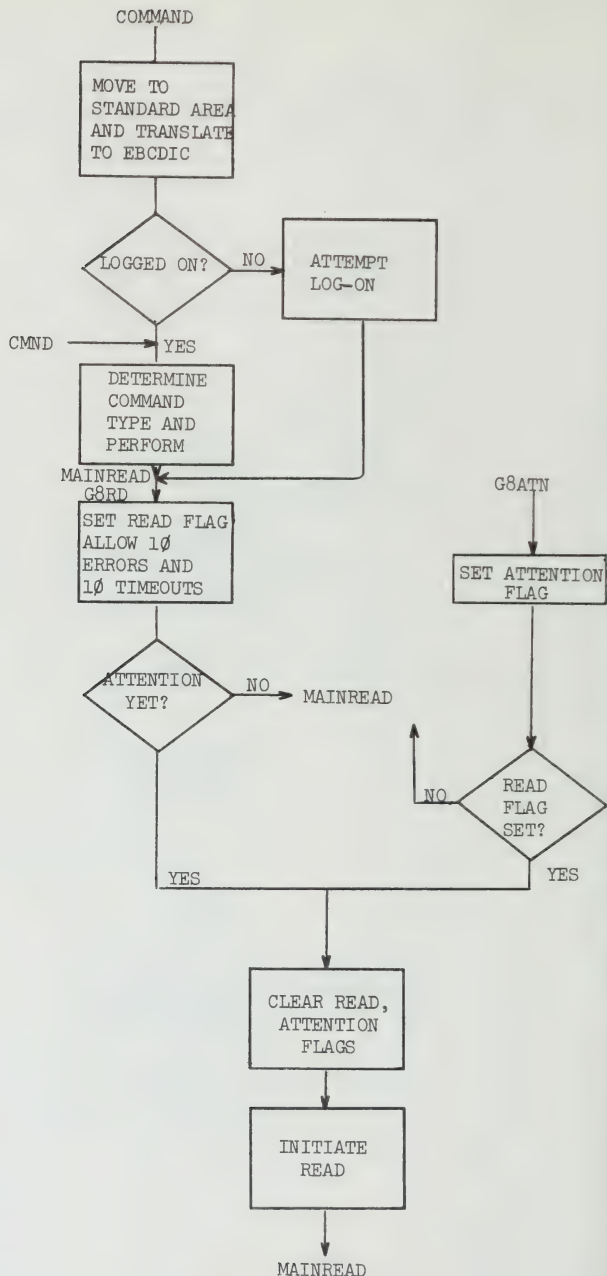
The length in PDP-8 words -1 is calculated from the `getmain` length and placed in the third control word (see above), and the `getmain` length is over-written with the linked list pointer (the last pointer being zero). After transmission, however, the third control word is used to recalculate the length of the buffer in order to `freemain` (release) it. This cumbersome method will shortly be replaced by eliminating the linked list, and using a table to keep track of records. This will allow the `getmain` and data transmission length to be zero.

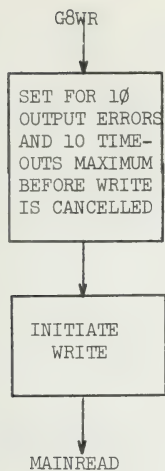
SPACT Flowchart



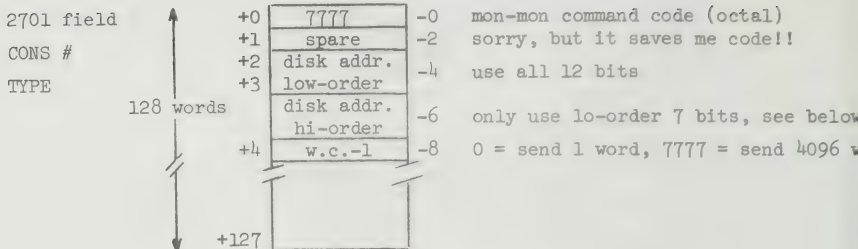




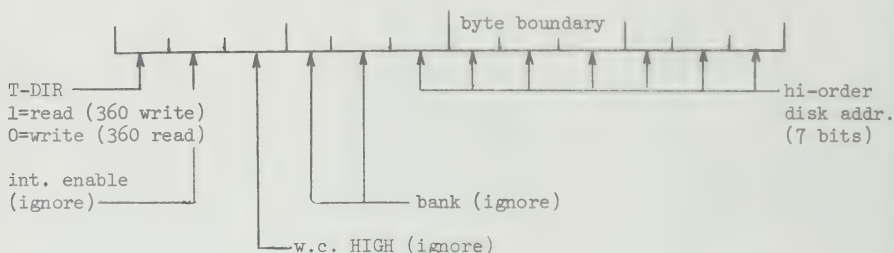




SIMULATED DATADISK REQUEST BLOCK



"hi-order" disk address (12-bit PDP-8 word)



Next record either way will be the data

3.1.5.2 . Changes to ITEM and COG

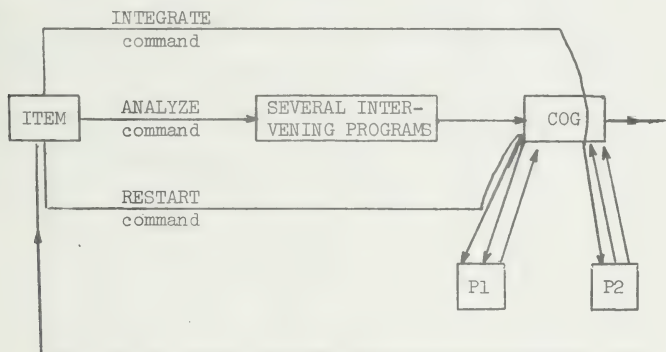
The following commands, corresponding messages, and bit settings have been added to ITEM.

TEST	TEST BIT ON Bit 7 set
NOTEST	TEST BIT OFF Bit 7 reset
LOOP	LOOP BIT ON Bit 6 set
NOLOOP	LOOP BIT OFF Bit 6 reset
RESTART <name>	RESTART INITIATED Bit 5 set
INTEGRATE <name>	INTEGRATE INITIATED Bit 4 set

The bits are contained in the parameter list provided by SPACT (list +8).

All programs in the simulation system have been modified so that they print diagnostic information only when the TEST bit is on. The LOOP bit is provided for testing an experimental version of the GLOBAL program.

Flow of control for the RESTART and INTEGRATE functions is outlined below.



The commands are used to bypass the rather complicated intervening programs in order to save computer time. The name which appears on these commands is the same as that which would appear on the ANALYZE command.

When COG receives a RESTART command, it assumes that the output normally given by previous passes is stored in a library file in the graphics filing system. The first six characters of each record stored (by COG) in this file consist of the name which appeared on the original ANALYZE command which caused them to be stored. The remaining two bytes of the record names (in hexadecimal) and the record contents are as follows:

0900	PW1	PW before execution of P1. Only the preset portion is saved.
0901	MX	As generated by P.E.E. Only preset portion is saved.
2E00	STUFF	Includes statistics such as number of equations, as well as S1, S2, and DIFFUN subroutines. All diagnostic information and other data from previous programs during an ANALYZE are stored in STUFF.

All addresses in STUFF are converted to displacements before it is written to the disk.

The final compiler stage for subroutines S1, S2, and DIFFUN in COMP has been modified so that the subroutines may be moved in core prior to the first time that they are executed. The output of P1 is also stored for later use by the INTEGRATE command. The last two bytes of the record names and contents are

0902	PWZ	PW as it appears after P1.
0903	EQN	Used by MATSET to
0904	VAR	calculate partial derivatives
0A00-0ANN	MATIN1	Matrix inversion subroutines.
0B00-0BMM	MATIN2	N+1, M+1, L+1, and K+1 are the
0C00-0CLL	MATIN3	number of segments in each
0D00-0DKK	MATMUL	subroutine.

Part of the output of P1 is subroutines compiled by SPARSE to invert a matrix. Due to changes in COMPIL, these subroutines may also be moved prior to their first execution. As pivot selection during matrix inversion is highly data dependent, the subroutines may have to be regenerated if some user parameters are altered, or if problem variables change in a way that causes a data exception in P2. This regeneration is the reason for the RESTART command, and may be initiated by the user as outlined above. Shortly, it will be possible to perform the restart automatically for data exceptions in P2. Manual restart for this situation would not be helpful as problem variables are not saved on the disk after COG terminates.

The INTEGRATE command is available for convenience to allow the user to repeat the solution of his equations. This could be caused by his desire to alter problem parameters (those not necessitating a change in pivot strategy), or a desire to examine output either not previously obtained (such as a plot) or forgotten.

The error messages RESTART NOT POSSIBLE and INTEGRATE NOT POSSIBLE are printed if required data for the specified function is not found to have been stored on the disk.

The changes in subroutine initialization for S1, S2, and DIFFUN are outlined in Figure 3501. The changes in the matrix inversion subroutine initialization is given in Figure 3502. Figures 3503 and 3504 give the calling codes for COMPIL (matrix inversion program generator) as they have not been previously documented.

*
*
*
*
*

SUBROUTINE INITIALIZATION CODE.

	DS	PD	
	USING	*,R15	
SPGN	STM	R14,R12,12(R13)	
SBAL	BAL	R12,SEGN1	
USE	DS	0H	
	DS	YL72	SAVE AREA.
DCONS	DS	A	DISP FROM "USE" TO CONSTANT TABLE.
DSUBS	DS	A	DISP TO SUBROUTINE ENTRY ADDR LIST.
DTLOCS	DS	A	DISP TO TEMP LOCS.
SCONS	DS	A	ADDR OF CONSTANTS.
SSUES	DS	A	POINTS TO SUBLIST.
NUMPARMS	EQU	10	SIZE OF PARAM LIST.
CPARMS	DS	(NUMPARMS)A	PARAM LIST FOR SUPPS.
SPARMS	DS	0C	COPY OF CALLING PARAMETERS.
TIMEVARS	EQU	**USE	
*T			
	DS	A	TIME.
PARAMS	EQU	**USE	
*G			
	DS	A	ADDR OF SPECIAL PARAMETERS ARRAY.
OUTPUT	EQU	**USE	
*DY			
	DS	A	ADDR OF OUTPUT ARRAY. L&M.
MVAR	EQU	**USE	
*Y			
	DS	A	ADDR OF M ARRAY.. INPUT ARRAY. 7XN.
LVAR	EQU	**USE	
*YL			
	DS	A	ADDR OF L ARRAY.
HINV	EQU	**USE	
*HINV			
	DS	A	ADDR OF VALUE: 1/H.
SPL	EQU	**SPARMS	
TLOCS	EQU	**USE	
STLOCS	DS	A	
C4096	DC	A(4096)	TO ADD TO ARRAY ADDRESSES.
BYPASS	DC	S(SPGN2)	NEW PLACE TO BAL TO.
SPGN1	DS	0H	
	L	R14,DCONS	FORM NORMAL ADDRESSES FROM DISPS.
	AR	R14,R15	THESE SUBROUTINES ARE STORED ON
	ST	R14,SCONS	THE DISK IN COLD STORAGE
	L	R14,DSUBS	SO THERE CAN NOT BE ANY REAL
	AR	R14,R15	ADDRESSES RETAINED. THIS
	ST	R14,SSUES	INITIALIZATION FORMS THE
	L	R14,DTLOCS	ADDRESSES ORIGINALLY EXPECTED TO
	AR	R14,R15	BE AVAILABLE WHEN THE COMPILER
	ST	R14,STLOCS	WAS WRITTEN. THIS CODE IS ONLY
	MVC	SPAL+2(2),BYPASS	EXECUTED ONCE.
SPGN2	DS	0H	ENTER HERE AFTER FIRST TIME
	USING	USE,R12	
	DROP	R15	
	ST	R12,2(R13)	
	ST	R13,4(R12)	
	LR	R13,R12	
	MVC	SPARMS(SPL),2(R1) COPY PARAM LIST.	
SEL	EQU	**SPGN	

Figure 3501

```

*
*      THIS IS THE INITIAL PART OF THE SUBROUTINE THAT IS
*      OUTPUT IN EACH CASE. IT HAS SUBROUTINE-LIKE THINGS IN IT
*      TO START OFF THE SUBROUTINE. IT ALSO HAS THINGS USED BY
*      COMPIL TO DO CODE GENERATION. ALL POINTERS AND WORK AREAS,
*      NEEDED BY COMPIL THAT MUST BE KEPT SEPARATE FOR EACH
*      SUBROUTINE. THE ORIGINAL COPY IS USED FOR SUBROUTINE ZERO,
*      SINCE NO CODE IS GENERATED FOR THAT SUBROUTINE ANYWAY.
*
SUB      DS      0D      ALIGN ANY DPL WD DATA AREAS.
NEXTSEG DC      A(0)    POINTS TO NEXT SEGMENT OF CORE FOR
*                               THIS SUBROUTINE (OR ZERO).
        DC      A(SEGLEN) LENGTH OF THIS SEG. (FOR FREEMAINING.)
        DS      XL10    FOR FILING SYSTEM...
        DS      0H      ENTRY POINT IN SUBP TABLE.
EP      USING   *,R15    ENTER SUBROUTINE HERE.
SUBREG  EQU      R15     REGISTER USED AS BASE IN THIS PROG.
USE     EQU      *       POINT OF ADDRESSABILITY.
        STM     R14,R12,12(R13) SAVE THE REGISTERS.
        B       SUB1     BRANCH AROUND CONSTANTS, WORK AREAS.
PARMS   DS      0F      INPUT PARAMETERS ARE STORED HERE.
PW      DS      A       ADDRESS OF PW ARRAY.
DY      DS      A       DY ARRAY.
DOUT    DS      A       DOUT ARRAY.
PARML   EQU      *-PARMS LENGTH OF PARMS.
CURRSEG DC      A(0) A(SUP) POINTS TO CURRENT SEGMENT BEING FILLED.
FREE    DC      A(0) A(SUP)
FREEND  DC      A(0)     AFTER CROSSING THIS, NEED NEW SEGMENT.
C4096   DC      A(4096)  HANDY TO COMPUTE ARRAY ADDRESSES.
STF2    DS      D       SAVE REGS HERE DURING COMPILING. THIS
STF2    DC      D'0'     WE CAN EXECUTE THE INSTRUCTIONS
STF4    DS      D       AS THEY ARE COMPILED. RESULTS ARE KEPT Y
*                               HERE.
CF2     DC      2H'-53'  -50 IS JUNK. ARRAY, INDEX OF VARIABLE Y
*                               CURRENTLY IN F2 (SO WON'T LOAD IT Y
*                               TWICE). NEG ARRAY # MEANS NEG OF
*                               VARIABLE IS F2.
NREGS   EQU      12      R1 TO R12 ARE FREE FOR USE.
REGTEL  DS      (2*NREGS)A CURRENT CONTENTS OF R1-R12.
SUB1    DS      0H      RESUME EXECUTION.
        MVC     PARMS(PARML),0(R1) COPY INPUT PARAMETER ADDRESSES.
        SDR     F2,F2     START WITH F2=0 SO WE CAN LOAD SINGLE Y
*                               PREC OPERANDS & USE AS DPL.
SUBL    EQU      *-SUB   LENGTH OF SUBROUTINE INITIALIZATION.
        DROP    SUBREG

```

Figure 3502:

*
*
* CALL COMPIL (OPCODE,ARRAY INDEX,ARRAY,SUBROUTINE #,PW)
*
*

* OPCODES ARE AS DEFINED IN INST.
*

* ALSO THE NEGATIVE OPCODES:

* -1 (LE F2, ?) DDR F0,F2
*

* -2 (LE F2, ?) LCR F2,F2 ?)
*

* -3 (LE F2, ?) DER F0,F2
*

* THE PART IN PARENTHESES WITH THE QUESTION MARKS IS
* CONDITIONAL. IF THE OPERAND IN QUESTION IS ALREADY IN
* THE REGISTER, THE CONDITIONAL PART IS OMITTED.
*

* ARRAY INDEX AND ARRAY ARE FOR RY INSTRUCTIONS. PP TYPES
* IGNORE THESE. NEGATIVE OPCODES USE ARRAY FOR THE LE F2.
* ARRAY NUMBER ARE:

* 0= PW ARRAY.
*

* 1= PW ARRAY.
*

* 2=DY ARRAY.
*

* 3=DOUT ARRAY.
*

* SUBROUTINE NUMBER INDICATES WHICH SUBROUTINE IS TO
* RECEIVE THE CODE. IF THIS NUMBER IS NEGATIVE, THEN
* OPCODE HAS THE FOLLOWING SPECIAL MEANINGS:

* 0= TERMINATE ALL SUBROUTINES.
*

* 1= INITIALIZE. INDEX=# OF EQNS., ARRAY=MAX. TYPE.
*

* 2= PIVOT DIVISION. INDEX=ROW#, ARRAY=COL#.
*

* 3= FORWARD ELIMINATION.
*

* 4= BACK SUBSTITUTION. INDEX=ROW, ARRAY=COL.
*

* PW MAY BE OMITTED WHEN ARRAY=1. USED TO EXECUTE OP.
*
*

Figure 3503

*			
*			
*			
INST	SDR	F0, F0	0
	NOPR	0	1
	LE	F0, 0	2
	ORG	*-2	
	LD	F0, 0	3
	ORG	*-2	
	STE	F0, 0	4
	ORG	*-2	
	STD	F0, 0	5
	ORG	*-2	
	NOPR	0	6
	MER	F0, F2	7
	AE	F0, 0	8
	ORG	*-2	
	MDR	F0, F2	9
	NOPR	0	10
	AD	F0, 0	11
	ORG	*-2	
	LD	F4, 0	12
	ORG	*-2	
CINST	LCDR	F0, F4	13
	LE	F2, 0	14
	ORG	*-2	
	STD	F4, 0	15
	ORG	*-2	
	SDR	F0, F4	16
	ADP	F0, F4	17
	STD	F4, 0	18
	ORG	*-2	
INSTL	EQU	*-INST	LENGTH OF TABLE.

Figure 3504

Introduction

The Illinois Graphics Computing System (IGCS) is a research and development program initiated in May 1971, involving the faculty and students of the Departments of General Engineering and Computer Science. It attempts to provide a low cost on-line, hands-on computing facility for in-class undergraduate graphics, simulation and design education. Physical hardware associated with the computing system represents two specific state-of-the-art hardware components: 1) a high-performance disk oriented minicomputer and 2) a high-speed electrostatic printer/plotter (EPP). The hardware configuration of the total system is shown in Table 1. This program is an attempt to build for the first time a high-performance, stand-alone, hardcopy graphics computing system for under \$50,000. Support for this effort has been derived from the Ford Foundation and Digital Equipment Corporation.

This report is one in a series of such reports dealing with research developments and performance of the system.¹

System Hardware Changes

Recently 4k of interleaved core was added to the system to bring its capacity to a total of 16k of interleaved core. An ECO hardware modification to the extended arithmetic element (KE-11A) corrects an earlier hardware fault.

* This section adapted from Progress Report #2, Departments of General Engineering and Computer Science, March 1972.

¹ See Illinois Graphics Computing System, Internal Progress Report #1, September 1971.

System Software Development

a. DEC-supplied Software

The following software systems are or will be soon supported in place of older versions: DOS V004A, Fortran-IV V003A, and Linker V007A. These represent the latest DEC releases. All systems programs have been modified in accordance with the monthly Digital Software News (L. Brown, M. O'Brien).

Recently acquired were system diagnostics on DEC tapes which are loadable into the system via the diode bootstrap loader. This has proved to be a considerable advantage over the previously used paper tape system since there is no high-speed paper tape reader in the IGCS system. The time saving for running regularly scheduled diagnostics has been large. (D. Hall)

b. IGCS Software

The line printer emulator program for the PDP-11, which outputs a disk file of printable ASCII characters to the Gould 4800 printer, has been used extensively despite a recurrent error which causes the printer to halt and the program to go into a loop. This problem is still being investigated. The time needed to form a line of print is dependent on the number of characters in a line; consequently, the Gould 4800 runs at variable speeds. The program can print up to 133 characters/line at a minimum speed of 540 lines/minute and up to 80 characters/line at a minimum speed of 900 lines/minute.

For the graphics project two programs were written and appear to be debugged. The first is a character generating subroutine which uses stored bit patterns to create ASCII characters in any of four orientations in a sector of a drawing. The second outputs to the Gould 4800 a disk file of previously generated bit patterns forming a drawing. (M. Clifton)

Two languages are currently being developed for use with IGCS: A Drafting Language (ADL) and Illinois Simulation Language (ILLISIM). ADL is a general purpose drafting language to facilitate the communication of graphical constrictions and annotations. Initial effort toward implementing ADL has centered on a plotting strategy for the raster format Gould EPP and the generation of geometric entities. Since the Gould 4800 EPP produces a smear if the paper is not advanced at least one scan line in a period of approximately 100 milliseconds, an initial strategy has been devised which places the bit maps for the entire drawing on disk and then dumps the drawing to the plotter. Moreover, to decrease the amount of random accessing of disk, the plots will be divided into sectors of about 1/2" (parallel to the scan lines). Disk buffers for the information necessary to generate a figure in the bit map are assigned to each sector. In total, the strategy entails a four-step process: 1) compilation of input, 2) placing information for generating the bit maps of figures into the sector file of the sector in which the figure first appears, 3) employing iterative techniques to generate the bit maps of the figures--one sector at a time, and 4) dumping the bit maps to Gould EPP. Each of these steps uses the output of the previous one as input. This strategy has been partially coded. (T. Runge)

The classes of geometric figures (in addition to annotations) of ADL will be limited initially to straight lines, circular arcs and elliptical arcs which correspond to circular arcs in the principal planes of an isometric drawing. The iterative techniques used to generate these are:

Metzger's algorithms³ for straight lines and circular arcs and the four center method (up to four circular arcs) for the restricted class of elliptical arcs. These figure generators have been coded and are currently being debugged. (D. Mueller)

The ILLISIM project encompasses the development of high-performance, digital simulation language tailored to the minicomputer environment. A stand-alone Adams method differential equation solver, based on DIFSUB⁴, has been written to solve several simple differential equation systems for ILLISIM using a subroutine overlay buffer in core to execute disk-resident program segments. A controller routine to interface with the output of the ILLISIM Compiler and provide for dynamic variable allocation is currently being written. (L Brown)

The design and structure of the ILLISIM compiler is complete and implementation is currently under way. The output of the compiler will interact with the routines above as well as the I/O routines. (W. Tam)

Additional IGCS software development covers two areas. First, the development of a software character generator to support the VT01 (Tektronix 611) direct view storage tube has recently been undertaken. Similar work which has been done for the PDP-8/KV-8 system at Carleton College under the name of COLPAC and development work at the University of Chicago for PDP-11/VT01 system is currently being studied. (D. Burn)

Second, a "Kluge" DOS monitor written by Gary Grossman of CAC is being modified to permit IGCS to load BASIC from DECTape. Under the DEC release paper tape system, BASIC loading is time-consuming and crashes DOS which must then be reloaded when BASIC is purged. (D. Mueller)

³ Metzger, R. A., "Computer Generated Graphic Segments in a Raster Display," Proc. SJCC, 1969, p. 161.

⁴ Gear, C. W., "Algorithm 407 for DIFSUB for Solution of Ordinary Differential Equations," CACM, 14, 1970, p. 185.

Cooperation

In an effort to interact with the rest of the user community, primarily on campus, IGCS facilities have been made available to a number of outside parties on an as-available, non-interfering basis. The Department of Electrical Engineering's Radiolocation Research Laboratory is currently utilizing IGCS for assembly language program debugging prior to trips to their PDP-11 field sites surrounding Champaign. In a cooperative project, Mr. Bruce Holecek, a graduate assistant in the Department of General Engineering and Mr. Ken Poole, a research associate with the National Clearinghouse for Correctional Programming and Architecture, the Department of Architecture, are initiating use of the IGCS system for on-line correctional facility cost estimating. The particular suitability of IGCS to their methodology should provide them with the opportunity to see their project reach a swift and successful conclusion.

The possible utilization of the IGCS for a remote job entry station experiment in ILLINET has been discussed informally with Dr. David Stonehill, Director of CSO and Mr. Clifford Carter, Research Engineer, DCS. The proposed part-time use of IGCS for this experiment would permit CSO to thoroughly evaluate the feasibility of a minicomputer-EPP system as an alternative to IBM 2780's.

Liaison with other minicomputer installations on campus, particularly the PDP-11 Educational Timesharing System in the Department of Computer Science and the PDP-11 Gould 4800 ARPA Network remote entry port in the Center for Advanced Computation has been maintained. Mutual discussion and assistance has centered around systems programming, software problems and maintenance.

Numerous interested parties from both within and without the University have inquired about the performance of our system and its applicability to their needs. In many cases, on-site tours of the IGCS facility have been conducted.

Performance and Maintenance Summary

Usage of the system has steadily increased since its introduction and is currently running approximately 40 hours per week with that time being distributed over the entire seven-day week and with the majority being during the first two shifts of the day.

Hardware problems with the IGCS facility have been fortunately very minor since the arrival of the system. The mechanical paper tape reader for the ASR 33 required read finger adjustment and was fixed by the personnel from DCS. Since adjustment, no further problems have been noticed. The stacker solenoid for the Card Reader failed and use of the card reader was limited until a new stacker solenoid could be secured. This failure had previously occurred in the DCS facility, and, based on that experience, no difficulty was encountered in repairing it. During the times that the stacker solenoid was out of use, the card reader was limited to experienced personnel since the card reader jam protect circuitry was also inoperative. The five-volt disk logic power supply upon start-up has on two occasions "protect-failed" leaving the entire disk logic locked in high. A temporary fix appears to be to shut the machine off to allow the power supply to unlock and then power up after waiting a few minutes.

It is our feeling that the in-house maintenance program is proving extremely cost effective and highly satisfactory. Maintenance and diagnostic work is carried out by two undergraduate students as electronic technicians in the Air Force. For further help we have been able to rely with complete confidence on the extremely helpful and very competent personnel in the Department of Computer Science. (D. Hall, G. Hodgson, W. Klemens)

Publications

Ruhl, R. L., "Illinois Graphics Computing System (IGCS). A Minicomputer Graphics System for Engineering Design Education," presented to Joint Midwest ACES/Simulation Councils Conference, October 21, 1971, Madison, Wisconsin.

Ruhl, R. L., "Applying Computer Assisted Instruction to Correctional Institutions: A Challenge to Engineering Educators." Presented to Joint Midwest ACES/Simulation Councils Conference, October 21, 1971, Madison, Wisconsin.

Clifton, M., Pleck, M. H., Ruhl, R. L., "Program LP: A Line Printer Emulator for the PDP-11/20--Gould 4800-11," submitted to 1972 Decus Spring Symposium.

MHP/pay
3/17/72

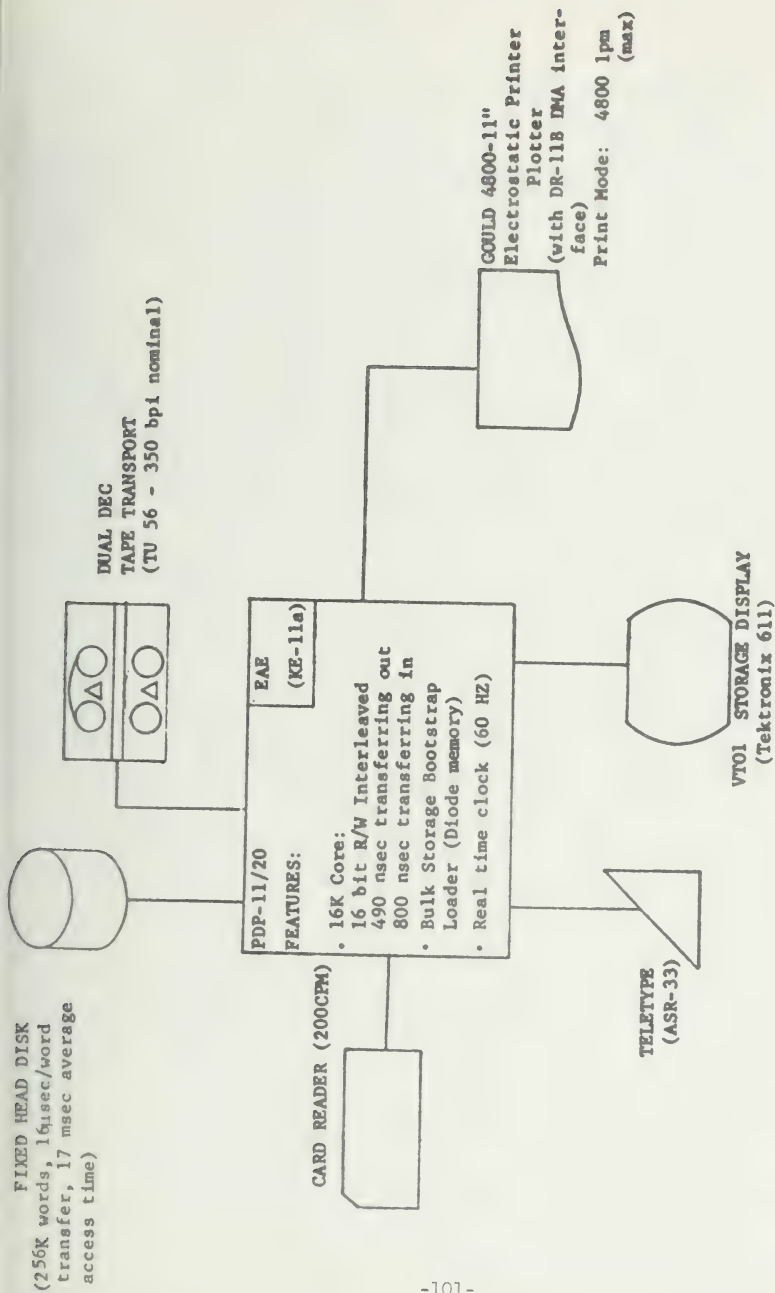


FIGURE 1: SYSTEM CONFIGURATION (3/1/72)

ILLISIM-E

The implementation of the ILLISM-E network compiler on the PDP-11/20 is being continued. Also, a specification of the network compiler at the macro level is completed so that the software package can be easily adapted to other environments. The structure and linkage of the network compiler to other packages are given in Figure 1.

With the arrival of version 003 of FORTRAN IV from DEC, more flexibility in the interaction between the peripheral devices and the compiler is available. Auxilliary programs like element/network dumps are being developed to aid the debugging process. A primary interfacing between the network compiler and the integration package is expected in the next quarter.

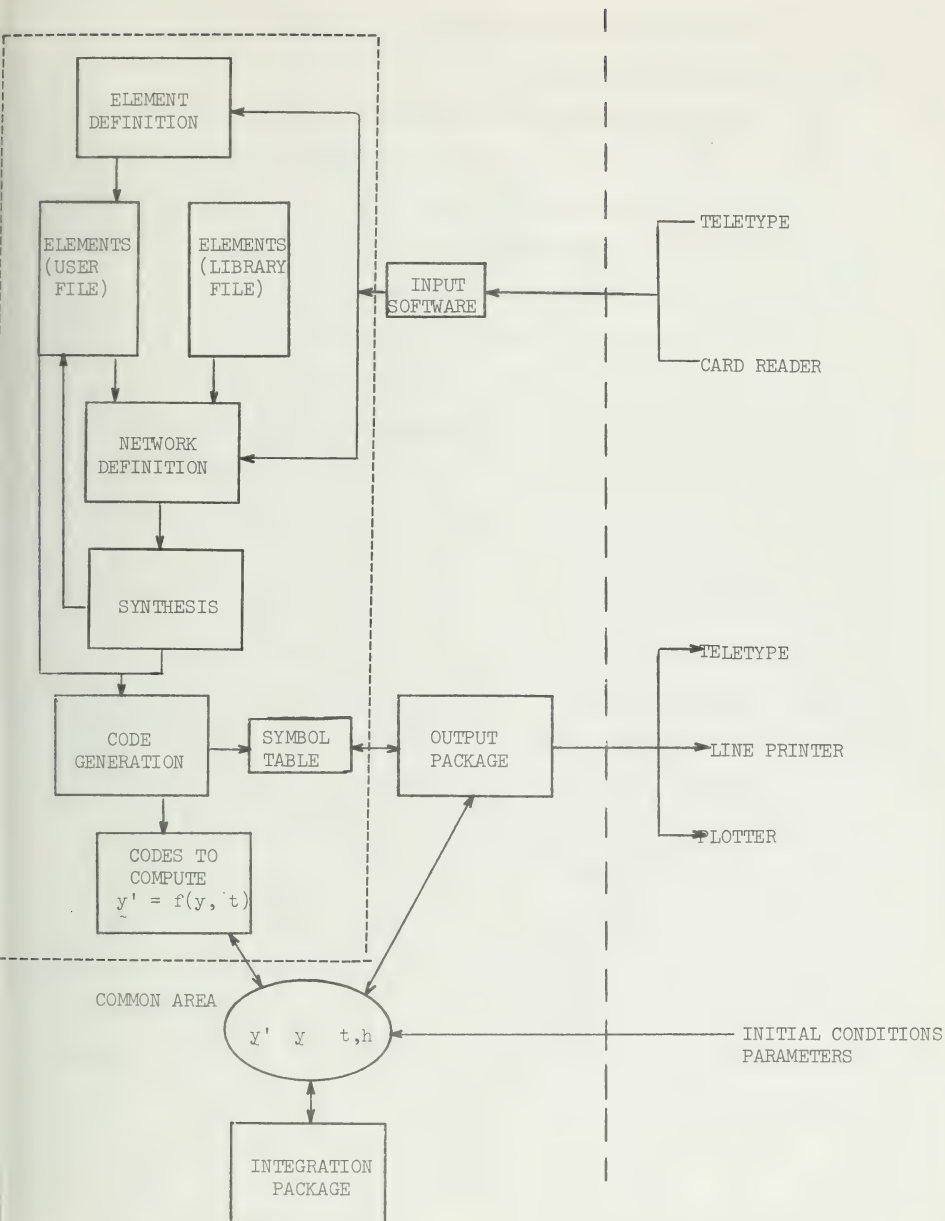


Figure 1

3.3 Graphical Remote Access Support System (GRASS)

3.3.1 Disk Monitor System (R. Haskin)

The unreliability of the data disk this quarter made it virtually impossible to use it for running the DMS. Consequently, all development work was done directly on Swaptape using the 'DMS' program. Several modifications to the system programs had to be made to allow them to run under the 'DMS' tape monitor head. These were

1. Modification of MON8I to use locations 7775-7777 for communication rather than 7765-7767. The old locations caused the monitor head to be modified with rather dire consequences. It was not possible to write SYSIO to fit on the page and not use 7765-7767.

2. LD8I, GLOAD, PTAP, and CTEK had to be modified to use the new locations for communicating with MON8I.

With the arrival of the SI disk, the modifications to the DMS to incorporate the new disk were made. Unfortunately, the SI disk is arranged into 128 word sectors, whereas the DMS requires 129 word blocks (128 data words, 1 link). The somewhat sloppy but quite usable solution decided upon was to use two sectors on the disk for each DMS block. Although this wastes almost half the disk, the remaining half (1024 blocks) is adequate for running the system.

The new monitor head is compatible with the 'DMS' tape monitor head, and the above mentioned changes to the 8-8I communications software carry over to the new disk system.

Conversion of Swaptapes to the new disk system follows one of two procedures:

1. If the Swaptape has no blocks used in the last SAM block, then
 - a. Using 'HELP', set the link word on the fourth SAM block to Ø. Also, allocate the last block in the last SAM block to file #1.
 - b. Use COPY to put the new version of EXIT on the DMS portion of the tape (also, VRFY, PTAP, LD8I).
 - c. Use XDUP to put the new monitor head (from another new disk Swaptape) onto blocks 400 and 401 of the tape.
 - d. Use XCOPY to copy the new version of SWAP onto the DECTape portion of the tape.
2. If the Swaptape has blocks used in the last SAM block (or if you are not brave enough to go through all the above steps), then:
 - a. Using any new system Swaptape and any blank tape, use NEWSYS to create a basic system on the blank tape from the Swaptape.
 - b. Swap on to the new disk from the newly created Swaptape.
 - c. Delete any unwanted files using PIP.
 - d. Use COPY to copy all files desired from the old version Swaptape. Since Swaptape formats are compatible from the old system to the new, COPY will work.

The only problem currently existing with the new disk system is that the DMS assumes that the TTY will be ready when it comes back from doing disk I/O. Since the new disk is twice as fast as the old, this is no longer true. Consequently, the .CD. messages come out rather garbled. Although this is surprising the first time it happens, it has no detrimental consequences, and a fix is not anticipated.

DEC's OS-8 operating system was purchased and received late this quarter. It is hoped that DMS will be replaced by OS-8 early next quarter to take advantage of the better assembler (with cross-reference listing) and improved editor. Implementation involves writing device drivers for the SI disk and Inktronic printer, and writing a program to convert DMS files on Swaptape (ASCII only) to OS-8 files. This will be done in the next two to three weeks.

3.3.2 Information Retrieval (J. Nickolls)

The current Information Retrieval (IR) system running under GLASP underwent some major changes this quarter due to the replacement of our disk. The new Systems Industries/Applied Magnetics disk is fixed head with 32 sectors of 128 words per track. There are 64 tracks, for a total of 2048 sectors or 262,144 words. The word transfer rate is 255 KHz, and the average access time is 8.7 msec.

Unfortunately, the disk is sector-addressable, whereas IR was designed for a word-addressable disk. (See DCS Report No. 409). The block allocation and deallocation schemes were completely redesigned, and more error checking and recovery procedures were added in line with our previous terrible experiences with an error prone disk. The disk layout was changed to allow storage of the resident system programs for quick system initialization.

The block allocation routines now use a bit map to keep track of the free blocks. This has the advantage that the blocks in a linked file may be selected sector-wise to minimize read/write time. It takes about 512 microseconds for one 128 word block to pass under the heads, which is enough time to get ready to read the next block.

If successive blocks in a file are picked so that

$$\text{Block}_{i+1} = \text{Block}_i + 2, \text{ mod } 32$$

then there will be enough time between blocks to issue the next read and avoid waiting the 8.7 millisecond average access time. The maximum transfer rate is, therefore, 16 blocks per revolution.

The bit map is stored on the disk in two parts, and is updated after a list of blocks (a file) is allocated or freed. Each piece of the map contains all 32 sectors by up to 32 tracks, so that a long file allocation only references the bit map currently in core.

Initial use of the system has shown the new disk to be virtually error free.

3.3.3 Monitors (J. Nickolls)

GLOAD/GEXIT

These two programs load and initialize the GLASP system, and are being completely rewritten to complement the new Information Retrieval System. The new load procedure is as follows:

1. Start MON8/I in the PDP-8/I unless it or ACID are already running. (See DCS Report No. 467)
2. Mount the GRASS load tape on unit 8, and if desired, a GRASS library tape on unit 4.
3. Start the DECTape monitor at 007600.
4. Type "GLOAD" followed by carriage return.
5. "OPTIONS:" will be printed out. Reply with a string of characters followed by a carriage return. The characters and their meanings are below:

Library Commands:

N--initialize the local library with no files.

T--load the local library from the GRASS library tape
on unit 4.

D--use the local library currently residing on the disk.

System Commands

S--load the disk with all system programs and load the
resident programs into core.

R--use the system programs already on the disk and load the
resident programs into core from the disk.

A--load ACID into the PDP-8/I. Either ACID or MON8/I must
be running in the 8/I.

If any characters conflict, the last one typed is used. No action is taken until the terminating carriage return is received. If no commands are given, R and D are assumed. At the completion of all commands, the GLASP system is started and a welcoming message will appear on the console.

The library tape has been restructured and is now file-oriented rather than being a disk image. This will facilitate online file manipulation between library tapes and the local Information Retrieval system in the future.

The tape directory is identical to the directory on the disk, except the block number entry is the first block number of the contiguous tape file. Also, the length entry is the actual file length in words, rather than the disk format. When the library is loaded from tape in GLOAD or when it is saved on the tape with GEXIT, the linked dish files are converted from/to contiguous tape files, and the directory format is changed.

All data read or written on the disk is checked by reading back and performing a word-by-word compare. Any bad transfers are signaled by a bell on the console and retried. The DECTape has proven itself to be absolutely error free, so no software checking is done for it other than a retry on parity error.

3.3.4 Remote Data Structure Utilities (J. Nickolls)

COMMUNE

Development of this package has necessitated some changes in calling sequences, which are documented below.

SENDP

It is now possible to give a picture sent to a terminal a name within the FORTRAN call:

```
CALL SEND(IPIC,ICODE,NAME,NAMLEN)
```

where IPIC, ICODE are as described in the previous quarterly report. NAME is an INTEGER*4 vector array containing a character string representing the picture name of the picture. The syntax is the same as for LSD--see DCS Report No. 466. Only the picture name is used; the library name and user name are ignored. The picture is saved under the local logon userid.

NAMLEN is an integer giving the number of characters in the NAME.

Note: NAME could be a literal string like 'PICNAM'.

It was found to be necessary to explicitly give the name length NAMLEN for all other routines using a NAME parameter. The new calling sequences are below. All other parameters and functions are as previously described.

It was found to be necessary to explicitly give the name length NAMLEN for all other routines using a NAME parameter. The new calling sequences are below. All other parameters and functions are as previously described.

```
CALL SAVE(IBLK,NAME,NAMLEN,IKEEP[,IPIC])
CALL SAVEP(IPIC,NAME,NAMLEN,IKEEP)
CALL FETCH(IBLK,NAME,NAMLEN[,IPIC][&STMT])
CALL FETCHP(IPIC,NAME,NAMLEN[,&STMT])
```

The calling sequences for CLEAN and CLEANP were omitted last quarter, so here they are

```
CALL CLEAN(IBLK[,IPIC])
```

CLEAN deletes a block from a picture. IBLK is the number of the block to be deleted. IPIC is the optional picture number and is defaulted to 1.

If IBLK = -1, all blocks of the picture are deleted, and if IPIC = -1, all pictures are freed of the block IBLK. If both are -1, all pictures are completely cleaned out, and the filing system is terminated.

```
CALL CLEANP(IPIC)
```

This performs the same as CLEAN, but an entire picture is deleted. IPIC = -1 deletes all pictures and terminates the filing system.

3.4 Computer Maintenance and Construction

3.4.1 Graphics-8 Hardware (C. E. Carter)

DATA DISK

Ten of the new amplifier cards have been put in the disk. Eight of these were put in the spare slot which is part of connector number 9. At the present time we now have available 72 tracks. Of these, only 48 are usable for disk storage as of this date.

NEW DISK

The Systems Industries disk has arrived. The necessary interfacing logic has been installed and checked out. The test program sent by Systems Industries does not work and an inhouse program has been written to check out the interface and disk. At this time the new disk appears to be working.

Two things need to be remembered when tieing into a DMØ1 multiplexor. The address accepted line from the DMØ1 must be terminated and the W-603 card must have pin V tied to the appropriate I.C. voltage (+5V.).

Drawings were made for this interface, but final versions remain to be produced.

COMPU TEK

All of the cards ordered for a fourth terminal have been delivered by the printed circuit lab. The six character generator boards (O, P, Q, R, S, T) have had all the components inserted except for the TIS58 gates. Three of the character generator boards have had the TIS58's

inserted and have been checked out. Board I has been wired and checked out. Boards W and V have been wired and are ready to be checked out.

Cables have been wired and installed between room 31J and room 115 DCL, and room 234 DCL. These have been checked out and may be used for demonstration purposes by moving a Computek terminal to either room.

One expanded pushbutton box for the Computek has been wired and checked. All of the necessary parts have arrived to build three more pushbutton boxes.

3.4.2 Equipment Maintenance Log Summary (H. Lopeman, R. Miller)

PDP-8

1. Bus that was common to switch register became intermittent due to bad solder joint. (Resoldered)

338

1. Bit 4 on pushbutton box not working. (Replaced and soldered broken wire in cable) A better clamping system has been used to hopefully keep this from occurring again.

DISK

1. Many problems with disk tracks. Since the data disk was returned the last time, it is extremely sensitive to temperature variations. (See quarterly report for October, November, December 1971, pp. 60 and 61.)

TELETYPE

1. Problem with PDP-8 TTY clock causing errors. (Adjusted clocks while using XOD routine and have had no complaints since.)

1. Position 14 of the accumulator failed. The repair was a cleaning of the board contacts.
- 2.. 2701 channel interface had a bad R-121 card. Also, a termination resistor was missing on one cable connection.
3. Memory test failed but was unable to find fault.
4. Contest II diagnostic test showed up a defective board in Rack E, Card slot B210.

PUBLICATIONS

- Gear, C. W. "High-Speed Compilation of Efficient Object Code,"
COMPILER TECHNIQUES, ed. B. W. Pollack, pp. 211-224, 1972.
- Koch, J. A. "Item Analysis," Department of Computer Science Report
UIUCDCS-R-507, March 1972; Master's Thesis, June 1972.
- Whaley, A. D. "A Failure-Tolerant System," Department of Computer
Science Report UIUCDCS-R-72-506, February 1972.

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

AEC REPORT NO. C00-1469-0206	2. TITLE 1st Quarterly Progress Report 1972
---------------------------------	--

TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

- ☐ c. Other (Specify) _____

RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

REASON FOR RECOMMENDED RESTRICTIONS:

SUBMITTED BY: NAME AND POSITION (Please print or type)

C. W. Gear, Professor
and Principal Investigator

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature

Charles W. Gear

Date

May 1972

FOR AEC USE ONLY

AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION
RECOMMENDATION:

PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

4. IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH: ILLIAC III
(Supported in part by Contract AT(11-1)-2118 with the U.S. Atomic
Energy Commission)

4.1 Introduction

A program of image processing experiments is being carried out. Our goal is to design a sight sensory system predicated upon parallel strategies for image processing. This development must of course evolve from our experience with the Illiac III system. The key property we seek is the ability of the system to learn rapidly from instructional interaction with professional personnel (not necessarily trained in computer technology). The validity of this orientation has been, we believe, vindicated by the rapid growth in significant image processing applications we can now attach.

Highlights of the program this past quarter include:

A. Interactive Picture Processing

1. Show-and-Tell has bedded down to being a workable, documented package for interactive picture processing.
2. An Atlas Extension of Show-and-Tell, described below, is still in the planning and early implementation stage.

B. Pattern Recognition

1. Covering Theory concept--extending to pattern recognition methods originating in switching theory, continues to be a very fertile ground for investigation. Interval Coverings have previously been described in our internal reports. Now Cartesian Covers, the latest extension of the covering theory methodology, have been introduced--accompanied by an operational program for their construction.
2. Vari-valued Logic, and outgrowth of the covering theory work which extrapolates from two-valued variables of switching theory to

quantized variables, has been introduced to provide a decision calculus for dealing with the pattern recognition problem.

C. Scene Analysis

1. Texture Analysis. A successful merger of concepts from interval covering theory (see B.1 above) and signal detection theory shows every promise of providing a first successful attack upon the problem of texture discrimination and analysis. Texture and color provide essential attributes for scene segmentation, i.e. the identification in the scene of candidate objects and their associated regions. Texture analysis is now generally recognized as a key obstacle to widespread application of image processing.
2. Shape Recognition, a necessary ingredient of the ATLAS system, has been successfully attacked in the work of Maruyama for simple two-dimensional shapes called "angularly simple." Extension to more involved forms is proceeding apace.
3. The ATLAS software package, an eidetic-based system for the processing of visual data, is emerging from the cocoon of its initial conception. Attempted applications to brain mapping and to aerial photointerpretation have stimulated much examination and extension of these concepts. A mathematical statement of the problem: how to use map information to interpret visual imagery, has been formulated.

D. Applications

1. Automated Cervical Smear Analysis

The prime objective was to examine the effectiveness of a parallel digital image processor in the analysis of cervical smear imagery. The emphasis here was not on the discovery and extraction of parameters of a malignant cell from a nonmalignant cell, but rather on the construction of algorithms for a parallel processor which permits

the computer to rapidly make sense out of the mess of cells and debris in the microscope field so that subsequent measurements (whatever they might be) are made on the correct objects: epithelial cell nuclei, for example, rather than clumps of white blood cells, cytoplasmic folds or locally similar-appearing phenomena.

A corollary investigation attempts to characterize the chromatin patterning of the nuclei, based on the texture analysis procedures mentioned in C.1 above.

2. Brain Mapping. A feasibility study of the automatic scanning of cell nuclei in brain tissue is being undertaken. Shape analysis of the nuclei is used to quantify transneuronal effects: cell dilation, cell altrification, etc. The tissue section is scanned under control of the automated light microscope, with digital control of the stage and focus.
3. Multi-Spectral Analysis. Imagery, both biological and remote sensing, where color plays a strong discriminatory role, are being examined by the prototype mechanism of using three-color separation negatives. Texture analysis techniques are being extended to these additional local (i.e. chromatic) attributes.

4.2 Interactive Picture Processing

4.2.1 Show-and-Tell

As a further step towards an independent ILLIAC III, Show-and-Tell was extensively operated on this quarter. Various routines were cleaned up, and new instructions to allow plane operations were added. By the end of the quarter, the PAX II subroutines AND, OR, EXOR, NAND, NOR, EQUIV, SHIFT, CLEAR, EQUAL were operating without any connection to the 360. LOADW and STOREW were also running, which allowed Show-and-Tell planes access to the digitized picture stored in the ILLIAC III core. Several demonstration programs were written in Show-and-Tell, which demonstrated that the speed of operation in local mode was sufficiently fast (4 ms, on the average, per plane operation). Our move from remote (360) to local (PDP8e) processing was thereby justified.

A minimal set of plane-operation primitives has tentatively been defined; they will be put into local Show-and-Tell, and experience will tell whether they are sufficient.

Remote Show-and-Tell and the 360 communications package suffered no change this quarter, even though changes were made to the 360 Operating System.

4.2.2 Atlas Image Deformation

Figure 1 shows typical (idealized) input. The map also is stored as another image, either in core memory or on film. Our intention is to interpret the input scene using the data structure implied by the map information. Initially, there is no spatial correspondence between the map and the image; they do not match.

Given both the map and the image, the operator selects the corresponding areas or windows that he wants to match. As in Figure 2, the map (Film 1) is linearly distorted until the center of the window matches the image (Film

2). The result is shown in Figure 3. If the window is sufficiently small, the map \rightarrow image transformation can be well approximated by a linear transformation, namely, the composition of translation, rotation, skew and possible rescaling. These functions are either already implemented or can be implemented in the available hardware.

More windows are matched, and a set of linear approximations is obtained. The centers of these approximations form a lattice which divides most of the map into disjoint triangles; an interpolated approximation of the map \rightarrow image transformation is then calculated for each triangle (Figure 4). (The union of these individual transformations is taken to be map \rightarrow image. The map is then transformed and overlayed in the image.) If necessary, more windows are selected until a satisfactory overlay is obtained.

This algorithm provides a natural manner of solution if we were to have a very flexible map overlay which we could stretch and compress until it fitted the image beneath. Interpolation functions have been found which should be able to approximate any degree of plastic (topology-preserving) deformation to any specified accuracy. These functions are third or fourth degree polynomials, but since they are applied in triangular patches and not over the entire area of the map, no great loss of accuracy results. They have also been carefully selected so that the transformations are at least first order continuous across triangle boundaries. Figure 5 gives a typical triangle.

DEC PDP-8/e SUBSYSTEM

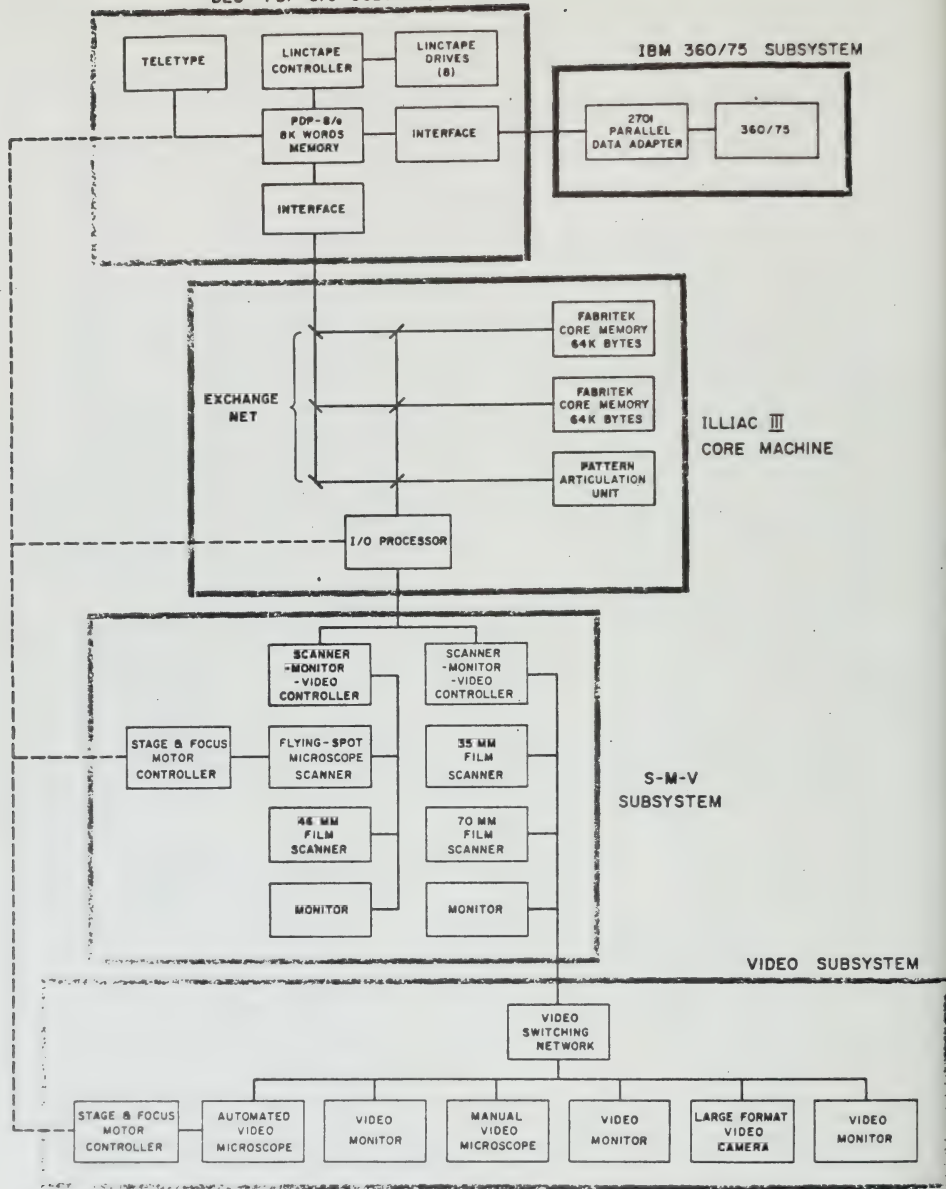
IBM 360/75 SUBSYSTEM

EXCHANGE NET

ILLIAC III CORE MACHINE

S-M-V SUBSYSTEM

VIDEO SUBSYSTEM



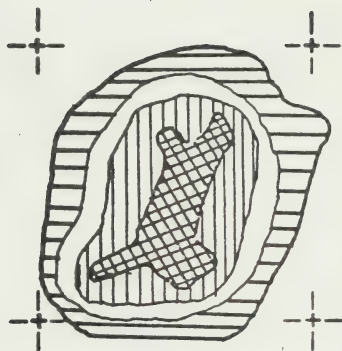
— DATA AND/OR CONTROL

- - - CONTROL ONLY

FIGURE 1. HARDWARE CONFIGURATION



MAP WITH FIDUCIAL MARKS
ON FILM 1



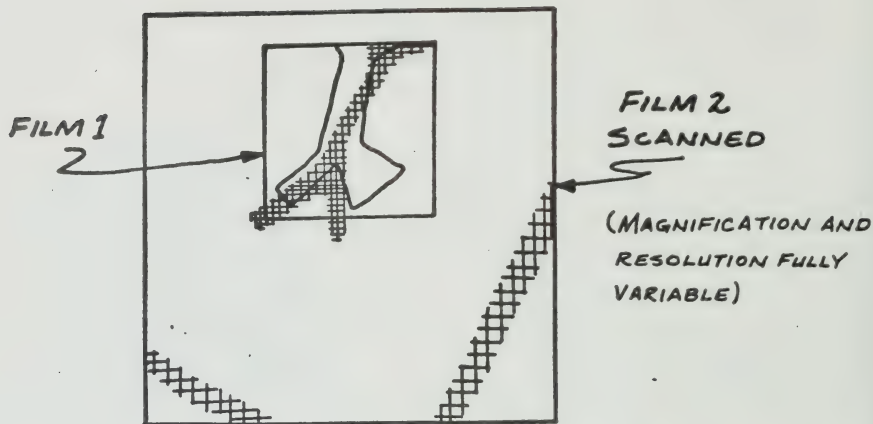
GREY SCALE IMAGE OF OBJECT
ON FILM 2 - WITH FIDUCIAL
MARKS
(NO CORRESPONDENCE WITH FILM 1)

1. USE FIDUCIALS TO IMPOSE COORDINATE SYSTEM ON MAP.

CURRENTLY STORE MAP AS A (MAX.) 256 x 256 x 4 BIT
GREY SCALE PICTURE. MAP CAN BE KEPT ON FILM.

INPUT

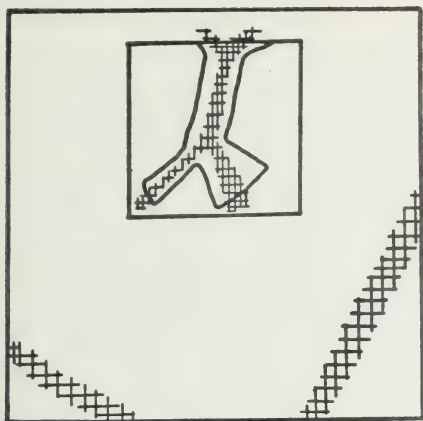
Figure 2



1. MAN SELECTS WINDOW OF MAP HE WANTS TO MATCH WITH IMAGE.
(CURRENTLY, MAP [FILM 1] HAS TO BE RESCANNEED FOR EACH NEW WINDOW)
2. MAN MATCHES CENTER OF MAP WITH IMAGE USING
'TRANSLATE' AND 'SKEW' OPERATIONS.
 - A) TRANSLATE ALREADY AVAILABLE AS A CHANGE IN SMV PARAMETER.
 - B) SKEW AVAILABLE AS SIMULATED (FOR $32 \times 32 \times 4$, 4ms COMPUTATION
 $< 1 \text{ MS } \frac{1}{10} \text{ TIME}$).

INTERACTIVE PROCESSING

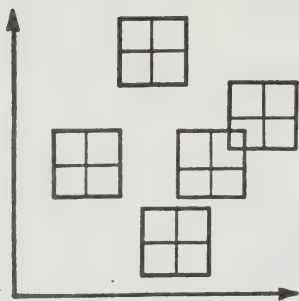
Figure 3



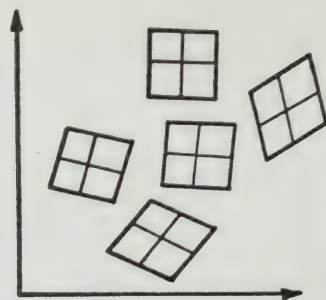
AFTER SKEW AND TRANSLATE
(CENTER OF MAP MATCHES IMAGE,
EDGES NEED NOT)

1. SIZE OF MAP WINDOW CHOSEN SHOULD REFLECT
CONFIDENCE OF MATCH- THE LARGER, THE BETTER.
2. CURRENTLY, HUMANS DO THE MATCHING- LATER,
WE MAY BE ABLE TO HAVE THE MACHINE DO IT.

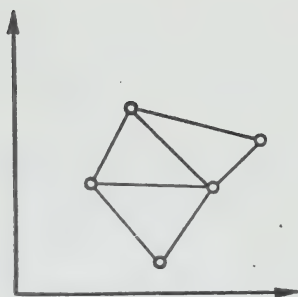
Figure 4



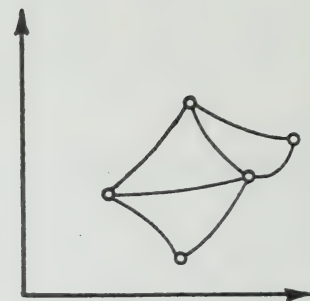
WINDOWS SELECTED IN MAP
(CENTER LINES REFLECT SKEW)



RESULTANT WINDOWS
AFTER MATCHING



TRIANGULARIZATION
OF MAP



RESULTANT

1. TRIANGULARIZATION IS USED TO COMPUTE TRANSFORMATION OF EDGES. THEN INTERPOLATION IS USED TO FIND TRANSFORMATION OF INTERIORS OF TRIANGLES.
2. THIS PROCESS CAN BE ITERATED UNTIL A SATISFACTORY MATCH OF THE ENTIRE MAP IS OBTAINED.
3. CURRENTLY, TRANSFORMATION IS DONE IN SOFTWARE. THESE PATCH-TRANSFORMS ARE CONTINUOUS ACROSS EDGES (0^{TH} AND 1^{ST} ORDER).

Figure 5

4.2.3 Scan/Display Device Development

Conversion of one 46 mm film transport to 35 mm form was carried out during this quarter. All optical and mechanical design was completed and 98% of the parts were manufactured. Installation of the conversion has begun.

The unit will handle standard 35 mm sprocketed film in the full frame (ne double frame) format found in most still cameras. Any length from a 3 exposure strip to a 400 foot roll can be loaded and examined (strips must be spliced in, or "clipped" in if they have sufficient non-image extra length).

All transport functions will be under PDP8e control. The film may be bi-directionally advanced by as little as .08 sprocket holes or .015 inches at a maximum of 200 steps per second. Using normal 8 sprocket per frame film, this leads to about one half second per frame advance, which is considered adequate. Film tensioning, clamp, and end-of-film flagging are automatic. Left and right end-of-film infra-red sensing will set a flag and disallow advance in the wrong direction. As there is a rather simplified control scheme for the film tension motors, no high speed slew or rewind is available; thus, maximum film speed will be limited to about 2 frames per second. This will probably instill caution in a user who wants to look at 200 feet of film in a session.

4.3 Parallel Processing Strategies

4.3.1 Texture Analysis

Using signal detection theory, a technique has been successfully demonstrated here which detects and isolates different textural regions in scenes of practical importance. To this purpose, a template is selected which defines a universe of patterns whose occurrences in the scene of analysis are to be studied for the purposes of discrimination between various textures. To quantify a judicious choice of size and shape of the template given a textural scene, we have resorted to the Time Series Analysis. Considering the textural scene as a two-way time series, traditional analysis can be carried out to determine the order of statistical dependence among neighboring picture cells. It is anticipated that a reasonable interpretation, using the Signal Detection Theory techniques employed earlier, can be obtained in terms of the Time Series Analysis.

(For further information see References [1], [2], and [3].)

4.3.2 Covering Theory

The work on interval and cartesian covers has reached the stage where it can be viewed as an independent discipline, a special branch of set theory. The basic problem here is that of expressing sets in terms of certain standard "building blocks" called complexes. The main effort in this quarter was devoted into experimental investigations, using operational program AQ-4C of the applicability of the developed concepts to selected problems of pattern recognition.

4.3.3 Variable-valued Logic

This work can be viewed as an outgrowth and a continuation of the work on covering theory on the basis of logic rather than set theory. Formally

it is an extension of the concept of many-valued logic. A paper which formally introduces a simple variable-valued logic system VL_1 and describes its application to picture description and pattern recognition has been prepared [4]. The system VL_1 unifies a number of problems which appear in such areas as pattern recognition, artificial intelligence, switching theory, graph theory, information retrieval, data reduction, etc.

4.3.4 PAX Language Support

Principal effort this quarter was directed toward the completion and mailing of the PAX Users Newsletter, Vol. 2, No. 1. This newsletter was sent to 45 people at 22 installations.

Also, seven requests for information concerning PAX were serviced.

4.4 Structural Inference

4.4.1 Scene Segmentation

An approach is being developed (1) to partition scenes into closed sets called regions and (2) to form composites of regions. Partitioning scenes into regions is performed in two steps:

(1) We assume that pictures are given as a set of picture points in the plane such that each point is associated with values of local attributes, e.g. its coordinates or a gray value. For N local attributes, the picture can be represented in a N-dimensional property space. In the property space, each picture point is represented by a N-dimensional vector whose n-th component carries the value of the n-th local attribute at this point.

(2) The property space is tessellated into N-dimensional unit cells. A region is supposed to be a set of neighboring picture points so that the density distribution of points in those unit cells containing the property space representation of the picture points is as uniform as possible. The uniformity of a properly normalized density distribution of points can be conveniently measured in terms of the entropy considered in information theory. We have stated algorithms that:

- (a) infer a metric in the property space for a given training set of pictures already partitioned into regions.
- (b) partition a picture into regions, using a previously established metric in the property space. The essential tool of this algorithm is a clustering technique using mode-searching and clustering in a minimal spanning tree.

The regions forming a partition of a scene can be represented as

nodes of a graph whose edges are labelled by attributes or relations and weighted by their values. A very efficient representation of such a graph is a vari-valued logic (VL) expression (see Section 4.3.3). Composites of regions can be formed by determining cartesian complexes synthesizing the vari-valued logic expression representing that graph. The algorithm A-8 that synthesizes VL-expressions allows control of the formation of cartesian complexes by imposing heuristic rules, permitting extensive experimentation on inferring composite objects. Furthermore, the representation of labelled, weighted graphs as VL-expressions is very efficient for applying graph transformation rules that have been developed earlier (see QPR April-June 1971, Section 4.3.1).

A description of these developments will be given in a thesis* to be issued in August 1972.

4.4.2 Shape Identification

Integral to the mechanization of using map information in the interpretation of imagery is the solution of the following problem of shape identification: we wish to match two complex figures, where each figure consists of many closed regions. One shape (figure) may also be a subshape of the other.

The above problem becomes fairly simple if each shape consists of a single closed region. However, if either one or both shapes is composed of many regions, then the problem can become very complex. In the latter case, it is not a simple problem to decide whether the relation graph of one shape is contained by the other relation graph, either as a

* Raulefs, P., "Methodological Aspects of Scene Segmentation," Department of Computer Science, University of Illinois, Urbana, Illinois, M.S. thesis, (in preparation).

complete subgraph or as a partial subgraph. It may also become necessary to apply graph transformations to facilitate matching. The two most significant transformations required are merging nodes and splitting of a node.

The following approach is being investigated for the solution of the above problem. The first step is to decompose the given two shapes S_1 and S_2 into smaller subregions using "neck," "tail," and "hair" properties. The second step is to construct relation graphs for these two shapes. The third step is a matching-into process: for each region $R_i \in S_1$, we select several corresponding regions of S_2 , denoted by S . For each $R_j \in S \subseteq S_2$, we again find corresponding regions of S_1 . By this approach, we establish regions of S which tentatively match to R_i . The fourth step is provided to determine graph structures, i.e., whether relations between two adjacent nodes of G_1 of shape S_1 hold between the corresponding pairs of nodes in G_2 of S_2 . Experiments are now in progress.

4.4.3 Structure Transformations

Our long range goal can be viewed as this: to design a system which can infer the syntactic and semantic structure of its visual environment from selected instances of objects and scenes from the environment. Implicit here is the prior existence of parallel processes for scene segmentation.

Three sub areas of the global recognition problem are identified: Picture Representation, Descriptive Pattern Analysis, and Processing Languages. The exclusive concern here will be with global aspects of processing, but we will try to use all local information and features present in the picture, [4], [5], [6], and suggest ways of incorporating

these into our global processor. Therefore, some proper preprocessing will be assumed on the picture before it is treated by this model. These processes typically consist of operations which can abstract low-level primitive objects as well as find relations between objects of the scene.

Fundamental to the development of higher level picture processing procedures is the creation of a suitable picture representation for algorithm and data. This representation should express the hierarchical structures of elements with attributes and relations among them. Basic to this model are a graph-structured data representation and graph-transformational parsing procedures. The graph-structural representation model has the following features which facilitate flexibility:

Subgraphs may be expressions of topological relations which permit a deformable model.

Attribute values associated with each object may be used to tie down the model to concrete instances.

Interaction or propagation of information between parsing levels, which is needed to identify objects in context, is readily expressible.

Descriptive pattern analysis attempts to build description of patterns.

We refer to the procedures which form the core of this analysis as parsing procedures. The central problems attacked by these procedures are the location, isolation, and identification of objects in a picture. The current lack of computational sophistication in attacking these problems is attested to by the elementary (by human standards) image processing that systems today are able to perform.

A labeled directed graph, whose nodes represent objects and whose edges represent arbitrary binary relations between objects, allow recognition procedures based on graph transformations. The recognition process can then be viewed as the application of replacement rules to graphs. The general structure transformation reference can now be specified by an ordered production

system whose entry points correspond to root graph operations. For example, the root operation of creating a branch, or merging two elements which may be dictated by the current picture segmentation strategy, specifies an entry point into the production scheme and thus determines possible graph transformations. These graph transformations can then specify actions calling for other root operation.

A thesis, to be issued next quarter, by John Schwebel [7] defines and develops the concept of graph structure transformations and their implementation in terms of a small number (basis) of fundamental transformations involving 2-4 nodes.

4.4.4 SOL (Structure Operation Language)

There has been a major progress in the development of this language. Currently it is embedded in PL/1, and its data structure is implemented using based and controlled variable facilities of PL/1. Also we have implemented a block structure compatible with PL/1 which consequently made it necessary that the related subset of statements be added to the language (GO, TO, DO group, etc.).

We also have developed a concept of graph input and output facility in the internal structural form to be implemented in our language. This is mainly to enable us to generate a graphic filing system operational with our language, which is inevitable in most applications of the language. We are planning to interconnect this picture processing facility with Show-and-Tell and Atlas on ILLIAC III, through an interface with 370/75.

4.5 Applications

4.5.1 Cervical Smears

The initial phases of this project were brought to a close last quarter. Some experimental results which used the interval covering technique were presented in a talk given at the IEEE-SPIE-Pattern Recognition Society Conference on Two-Dimensional Signal Processing in Columbia, Missouri. A paper on the same subject was written and submitted to the IEEE Transactions on Computers. In addition, a thesis titled "Parallel Image-Processing for Automated Cytology" (by John Read, to be issued as a Department of Computer Science Report) was brought to a state of near completion.

4.5.2 Brain Mapping

In research concerning the study of neuro-interconnectivity relations, a primary problem is the vast amount of time needed to get meaningful data from the counting of neural cell slides. This operation involves the occupation of trained individuals working with a microscope at oil immersion power.

The purpose of this research is a study of the feasibility of using the ILLIAC III system to scan, count, and identify neurons from biological slides of cat brain limbic system. If such a project proves to be feasible, then an attempt will be made to use the system to perform intercomparative data gathering operations on actual brain slice slides whose neuron count has already been determined. A comparison of the machine count and the manual count will then be made, using the manual count as the standard for accuracy.

At the present time this project is in the feasibility study stage.

4.5.3 Cytospectrometer

During this quarter, small advances were made in further construction and initial testing of the Taylor cone gun, thermal pump, and beam transport tubes.

The thermal pump scheme was used to see if a Taylor cone could be formed using water. Previous attempts at cone formation with static water levels at the needle tip met with near zero success because there was no liquid flow to replace material drawn off at the tip. This doubtless led to collapse of the cone in the absence of increasing voltage. Hence the pump was added and the test run again. This test was somewhat more successful as the cone (or something like it) formed and remained "stable" for perhaps a second at a time before lapsing into wild instability--followed by periodic formation and collapse as the pump caught up with the flow demand and then fell behind.

The chief problem appears to be with viscosity. Taylor cones are most often observed with rather viscous fluids such as paint or glycerin; water and blood have much lower viscosities and are thus probably much more difficult to stabilize into a cone. Surface tension is also a major factor but is not speculated upon at this time. A stable cone is the goal, but it must be stable with liquids compatible with biological samples.

Conversion of a lathe for generating the spiral conductor patterns on beam transport tubes is nearly complete. A pair of reduction wheels suitable for a 4 mm tube is being installed, and a 4 mm chrome-coated tube with quadrupole conductors should be finished in May. The tube will be about 200 mm long and have a conductor lead of 1 turn in 5 diameters. Testing of its ability to focus 150 mm charged water droplets should begin in June. Reduction wheels for tubes down to 1 mm O.D. have been calculated and are easily manufactured.

4.6 Computer Systems Support

4.6.1 IBAL Assembler

There has been no major development this past quarter. However, we have spent some time to optimize the representation of data structure in memory. Arrangements have been made to tackle the final stage of implementation this summer.

4.6.2 PAU Development

The PAU Simulator was completed and debugged. A simple interpreter was written to facilitate the use of the Maintenance Processor to check out the Iterative Array hardware. Checkout is continuing.

4.6.3 Processor Intercommunication

The interface between the Maintenance Processor and the Exchange Net (and consequently the Fabritek fast core) has been completed and is operational.

The LINC tape controller design is completed, back panel wiring lists prepared, and 75% of the boards required have been wired and checked out.

4.6.4 Scanner/Monitor

During the period of January to March, the SMV was operational approximately 278 hours.

Production & Demo	approx. 128 hours
Corrective Maintenance	approx. 20 hours
Preventative Maintenance and Modification	approx. 100 hours
UNLOGGED USEAGE	approx. 50 hours

During this period the microscope scanner was used quite extensively. However, no attempt is made to differentiate between film scanner and microscope useage.

During this period, a minor wiring error, of long standing, was discovered. It became evident only when very small scanned areas were magnified highly.

Adjustment for maximum gray-scale resolution has been rewarding. A full sixteen shades of gray are readily discernable. However, most shades of gray do not fall exactly on a specific digitized value of gray, and the evidence of change of gray level is more evident on a quantitative evaluation across a photographic gray scale, for instance, than by a single value of light transmission through a given film density. Non-homogeneity of film density rather aggravates this condition.

It has also become apparent that certain gray-scale values in the mid-range have an obvious tendency to generate more "noise" than any of the others.

The video-digitization section is almost ready for hook-up and debugging. The final connection for power is awaiting delivery of certain hardware and the completion of the back panel wiring of the analog section.

No work has been done on the Channel Interface Unit sections during this period.

The existing By-Pass unit has been modified to manually generate different unit addresses. Primarily this is to allow access to any of the large core-memories.

REFERENCES

- [1] Hannan, Multiple Time Series, John Wiley & Sons, Inc., New York, 1970.
- [2] Box and Jenkins, Time Series Analysis - Forecasting and Control, Holden-Day, Inc., New York, 1970.
- [3] Bhattacharya, P.K., "Order of Dependence in a Normally Distributed Two-Way Series," University of Arizona.
- [4] R. S. Michalski, A variable-valued logic system as applied to picture description and recognition, Proceedings of the IFIP Working Conference on Graphic Languages, May 22-26, 1972, Vancouver, Canada.
- [5] K. Maruyama, "Shape Detection," DCS thesis in progress.
- [6] S. N. Jayaramamurthy, "Texture Analysis," DCS Thesis in progress.
- [7] J. C. Schwebel, "A Graph Structure Transformation Model for Picture Processing," Ph.D. Thesis, U of I, May, 1972.

4.7 BIBLIOGRAPHY

4.7.1 Outside Lectures:

R. S. Michalski: Lectures on Variable-valued Logic and Its Applications to Pattern Recognition, Picture Processing and Artificial Intelligence.

February 7, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada

February 9, Department of Applied Analysis and Computer Science, University of Waterloo, Ontario, Canada

February 10, Department of Electrical Engineering, McGill University, Montreal, Canada

February 14, Project MAC, Massachusetts Institute of Technology, Cambridge, Massachusetts

February 16, Center for Information Research, University of Florida, Gainesville

4.7.2 Logic Drawings Issued

The following new logic drawings have been drawn and issued during the past quarter:

TP Control Logic	42 drawings
TP Control Logic	18 drawings

4.7.3 Engineering Drafting Report

During the past quarter a total of 313 drawings, including new logic drawings, drawing changes, layouts, flow-charts, theses, report drawings, and drawings related to the Opto/mechanical design of the ILLIAC III project have been processed by the 2118 drafting section.

4.8 ADMINISTRATION

4.8.1 Personnel Report

Senior Staff

Professor Bruce H. McCormick - Principal Investigator
Assistant Professor R. S. Michalski

Professional Staff

Robert C. Amendola
Richard T. Borovec

Research Engineering Assistant

S. Paul Krabbe

Electronic Engineering Assistant

Joseph V. Wenta

Digital Computer Technician II

George T. Lewis

Drafting

Stanislavs Zundo

Research Assistants

Walter Donovan
S. N. Jayaramamurthy
Ahmad E. Masumi
Kiyoshi Maruyama
Frank Murakawa
Peter Raulefs
Kuo Wen

Secretarial

Mrs. Star Starnes

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO. C00-2118-0033	2. TITLE Illiac III Quarterly Progress Report
--	--

3. TYPE OF DOCUMENT (Check one):

☒ a. Scientific and technical report

☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

☒ a. AEC's normal announcement and distribution procedures may be followed.

☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.

☐ c. Make no announcement or distribution.


5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

Bruce H. McCormick
Professor of Computer Science and Physics

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature 	Date May 19, 1972
--	----------------------

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

☐ a. AEC patent clearance has been granted by responsible AEC patent group.

☐ b. Report has been sent to responsible AEC patent group for clearance.

☐ c. Patent clearance not required.

5. CENTER FOR ADVANCED COMPUTATION

Report Summary

All of the progress reports in this section, except for the NARIS and IRIS projects, were sponsored under ARPA contract DAHCO4-72-C-0001 entitled "ILLIAC IV Applications Research at the Center for Advanced Computation, University of Illinois at Urbana-Champaign." The principal objective of this program is the development and testing of numerical techniques and software systems for use of ILLIAC IV over the ARPA Network. This is being accomplished through activities in the following areas:

1. Development of numerical techniques suitable for parallel processing in the following areas:
 - a. Computational Methods in Linear Algebra.
 - b. Linear Programming.
 - c. Ordinary and Partial Differential Equations.
 - d. Time Series Analysis.
 - e. Quadratic Assignment Code.
 - f. Graphics.
2. Development of ARPA Network facilities and associated systems consisting of:
 - a. ARPA Network Terminal System (ANTS).
 - b. Center Graphics Support.
 - c. Network Graphics Effort.
3. Development of a large scale applications system, dealing with input-output economic modeling, utilizing ILLIAC IV algorithms and several ARPA Network facilities.
4. ILLIAC IV Language Development.
5. ILLIAC IV Information Retrieval and Statistical System (IRSS).

6. ILLIAC IV Image Processing.

7. ILLIAC IV Education.

The NARIS and IRIS projects are concerned with design and development of geographic information systems for Illinois.

5.1 ALGORITHM DEVELOPMENT GROUP

5.1.1 Introduction

The main objective of this group is the development of numerical techniques that are most suitable for parallel machines, namely the ILLIAC IV. The research program can be divided into the following areas:

- (a) Computational Methods in Linear Algebra
- (b) Linear Programming
- (c) Ordinary and Partial Differential Equations
- (d) Time Series Analysis
- (e) Quadratic Assignment Code
- (f) Graphics

In developing any parallel algorithm, we first implement the existing serial algorithm on the B6700 (in Algol). Once the programmer is familiar with the computational characteristics of that routine, the parallel algorithm is then designed. In many instances, the parallel algorithm is substantially different from the serial one, in which case the parallel routine is written on the B6700 in Algol to check its validity, accuracy, rate of convergence, etc. The last step is then writing Glypnir or ASK code, and finally debugging it on the B6700 simulator (SSK). Due to the slow speed of the simulator, many codes cannot be debugged completely. Some time consuming codes have been coded in Glyplit (Glypnir to PL/1 translator developed at RAND) for fast checking of their logic only.

The group has adopted a verification process of the algorithm in which each member tries to use and test algorithms developed by other members. This leads to the discovery of well-hidden bugs and helps in pointing out areas of needed standardization and documentation, as well as improving the codes being reviewed. A listing of the Center's Subroutine library is now available, [1], which identifies available debugged algorithms, describes their function, the language in which they are written, and the author.

5.1.2 Computational Methods in Linear Algebra

5.1.2.1 Solution of Systems of Linear Equations

(a) An ASK program was developed to solve systems of linear equations $Ax = b$ for many right hand sides, where A is a banded positive definite symmetric matrix. Work has been completed on the algorithm to handle matrices of band-width < 64 , and the algorithm for bandwidth < 127 is currently in the debugging state.

Enroute to the ASK algorithm, two routines in Algol were extracted from Wilkinson and Reinsch's Linear Algebra, [2], for incorporation in the group's subroutine library:

- (i) a routine which decomposes the banded matrix, and
- (ii) a routine which given the decomposition, proceeds to determine the solution vectors for many right hand sides.

Documentation on the Algol program is completed, and the ASK programs will be documented upon their completion.

(b) A program for the decomposition of a matrix into the product of lower and upper triangular matrices for the solution of linear equations or matrix inversion, [2], has been completed in ASK for non-core contained matrices (order ≈ 1000). The program has been debugged as far as possible but the slow speed of the simulator and lack of satisfactory input/output routines have precluded further checking until the machine becomes available. A document is forthcoming. A Glypnir program for solving complex systems of linear equations, order 64 , by the LU decomposition has also been completed, [2,3,4].

(c) An ASK program of the Householder triangularization, [3], of a matrix and back-substitution for solving core-contained systems of linear equations (order ≈ 300), is now in the final stage of debugging on the SSK simulator. The routine consists of seven external subroutines and a master program, all of which are checked out independently. The task presently in progress involves combining individual subroutines, several at a time, for checking on SSK without exceeding its capabilities.

Householder's Modification method for matrix inversion has also been coded and debugged in ASK but only for matrices of order 64 or less since the number of operations is higher than that of the LU decomposition or Householder triangularization. The method seems to be suitable for sparse matrices. A document describing both Householder's triangularization and the Modification methods is forthcoming.

(d) The conjugate gradient method for solving linear equations $Ax = b$, [2], has been coded in Algol to gain experience in the technique. Glypnir coding is proceeding but has been somewhat hampered by lack of facilities in the language, namely the inability to call the subroutines as parameters. This algorithm requires as an input a routine to compute Ay and $A^t y$ for a given y rather than an explicit representation of the matrix A . This should prove to be very fast in cases where A is sparse and of regular structure.

5.1.2.2 The Algebraic Eigenvalue Problem

(a) Two ASK programs for parallel Jacobi and Jacobi-like algorithms, for real symmetric and real nonsymmetric matrices of order 64 or less, have been completed, and a document has been published [5]. An improved parallel Jacobi algorithm has been developed [6], and an ASK program has been completed and documented [7]. Several experimental Algol and Glyplit programs have been written to explore the possibility of improving the convergence of already published Jacobi-like algorithm [5,6].

(b) The QR algorithm for finding the eigenvalues of symmetric tridiagonal matrices has been coded in Glypnir.

The QL algorithm [2] for finding the eigenvalues and eigenvectors of symmetric tridiagonal matrices has been coded and tested in Algol and being programmed in Glypnir. A generalization of the bisection method together with inverse iteration have been programmed and tested in Algol. Both routines are being programmed in Glypnir to produce a parallel algorithm, [8], with higher efficiency than the inherently serial QL algorithm.

(c) The conjugate gradient algorithm for finding the largest and smallest eigenvalues and the corresponding eigenvectors of the generalized eigenvalue problem $Ax = Bx$ where A and B are symmetric positive definite matrices, [9], has been coded and tested in Algol and will be coded in Glypnir with high efficiency of parallelism. As mentioned earlier, Az and Bz can be generated implicitly for any vector z , when A and B structured, sparse matrices (which is the case in most practical applications).

(d) The accelerated power method for finding the largest eigenvalue and the corresponding eigenvector has been coded and tested in Algol and Glyplit. This method, mentioned by Wilkinson [3] (ad hoc shift of origin) for man-machine interaction, was modified so that it would not require interaction unsuited for the ILLIAC IV. A variation of the above power method now being tested, where the L_2 norm is used instead of the L_∞ norm, is just as fast on the ILLIAC IV and more predictable, hence more suited to unmanned operations.

(e) The QR algorithm, with origin shift for the solution of eigenvalue problems, [2], has been coded in ASK for non core-contained matrices and is being debugged. The reduction to upper Hessenberg form using Householder's transformations, [2], of non-core-contained matrices is almost totally debugged.

An ASK program for reducing core-contained real matrices to the upper-Hessenberg form using elementary stabilized transformation [2] is completed. Two documents containing ASK codes for the QR algorithm and Householder reduction to upper-Hessenberg form for matrices of order 64 or less are available [10,11].

(f) An algorithm for establishing error bounds on the complete eigensystem of core-contained (order ≈ 90) diagonalizable matrices [12] has been coded in ASK and is in the debugging stage. This should reveal some of the favorable aspects of parallel computation.

(g) A numerical method for obtaining the eigenvector solution of the non-linear matrix Riccati equation has been programmed and fully tested in Glypnir, and a complete document is available [3].

(h) Work has begun only recently on the development of an eigenvalue algorithm for obtaining all eigenvalues and vectors of sparse matrices. As a first step such an algorithm was developed for matrices whose Gerschgorin disks are all isolated. The resulting parallel algorithm has for its basis the work of Varga and his associates on the best isolated Gerschgorin disks [13,16]. The Algol program has been extensively tested; results obtained compare favorably in accuracy with the QR algorithm. Work continues on the general case where the Gerschgorin disks cannot be isolated by diagonal similarity transformations.

5.1.3 Linear Programming

(a) The goal of the initial LP codes is a solution capacity of 16,000 row problems with a potential for expansion to 64,000 row problems. Codes initially available will be PRIMAL, a modification of the revised simplex method utilizing a product form of the inverse, and INVERT, a reinversion program based on the Hellerman-Rarick preassigned pivot procedures. Supporting these computational portions are the data massaging SETUP and SETDOWN codes which create initial files and produce result-files, respectively. The LOADER, a B6500 program, accepts the data in MPS/360 form and after sorting passes this data to the ILLIAC IV disk. SETUP then utilizes the ILLIAC to convert the values to suitable word formats, classify columns and rows, scale these columns and rows, and emit those files necessary for PRIMAL and INVERT. SETDOWN picks up at the termination of the PRIMAL-INVERT symbiosis and rescales and edits the output, still using ILLIAC. An UNLOADER will eventually be written to replace the internal (numeric) with the original external (alphanumeric) labels and to prepare printer output.

The Linear Programming group, presently staffed by fewer than two full-time staff members, progressed steadily through INVERT coding. INVERT has been divided into three sections:

- (i) the creation of an implicit in-core matrix representing the basis vectors,

- (ii) the identification of the pivot sequence for the new Product Form of the Inverse (PFI),
- (iii) the creation of the ETA files representing the PFI.

The first two sections have been coded and simulated; while their lack of parallelism makes them less than satisfactory, they are the best algorithms identified so far. The third section appears to make very good use of the parallelism of the machine and should be simulated by the end of the present quarter.

PRIMAL has been coded and debugged as far as the simulator could allow. The major revision being considered is the inclusion of debugging and diagnostic routines. A lack of response to our letters requesting information from NASA/AMES Institute for Advanced Computation (ascribed to a shortage of personnel at IAC) has postponed a rewrite of our I/O utilities and the development of these diagnostic routines. Information obtained at the March 1972 user's conference at Monterey suggests that ILLIAC IV will not begin to support LP's I/O requirements for some time.

The SETUP routines are nearly completed and, with the current resolution of the last section of INVERT's flowcharts, should be quickly disposed of. Documentation of all coded routines are available and will be combined into one document describing the whole system once unresolved issues about ILLIAC IV are settled.

(b) Preliminary investigation of methods to reduce storage and computation time for large Linear Programming problems have begun. This would be achieved by performing all matrix operations (e.g., calculating Ax and $A^T y$ given x, y) implicitly without ever generating and storing the matrix A . Used with a matrix generator, significant savings may be obtained.

5.1.4 Ordinary and Partial Differential Equations

(a) A preliminary study for an ordinary differential equations solver for ILLIAC IV has been completed and a document is available [17].

(b) Two Glypnir codes have been completed for the alternating direction implicit iterative technique (ADI, 64 rows), and the block-Jacobi method (128 rows) for the solution of second-order elliptic partial differential equations. A document describing the block-Jacobi method on the ILLIAC IV is forthcoming.

(c) A Glyplit program has been written which performs simultaneous fast Fourier transforms of real functions in all enabled PE's. At present, the program is core-contained. This program may be useful when solving partial differential equations by means of a change of variables that can be carried out by the fast Fourier transform [18].

(d) A Glypnir program implementing J. Son's (Department of Chemical Engineering, University of Illinois at Urbana-Champaign) solution to the problem of flow around a cylinder has been coded and debugged on the simulator. It is estimated that a 50% reduction in run time can be achieved by coding in ASK the section which solves Poisson's equation iteratively.

Direct methods for solving Poisson's equation appear to offer additional reductions in run time. Hockney's direct method has been implemented in serial form in Algol. This algorithm was compared to a serial SOR and found to be about twice as fast for those cores tested.

A direct algorithm for solving Poisson's equation in an irregular domain using a Hockney type formulation is being developed. A serial iterative algorithm has already been developed in Algol for these irregular domains.

Various boundary conditions are also being tested in an attempt to better model the physics of this fluid flow problem.

(e) A computational technique for high subsonic compressible inviscid flow past a circular cylinder has been implemented serially on the CDC 1604 computer; a Glyplit code will be written soon. Boundary conditions are chosen for maximum utilization of the processing elements. Procedures for varying grid sizes without changing the PE instruction stream have been developed. The calculated results compare well with

those obtained by Holt and Masson [18] by the method of integral relations. A document has also been completed [19] in which the relevant and practical criteria required for the numerical calculation of different types of gas dynamic flow fields are discussed.

(f) Glypnir programs have been written to solve a set of differential equations used in atmospheric dynamics to describe fluid convective motion induced by the buoyancy force and heat transfer associated with the motion. Time comparisons have been made between these programs and CDC 6400 FORTRAN programs for a 64×17 mesh. The ratio is about 1:110 when successive overrelaxation method is used in solving Poisson's equation. To improve this ratio, the kernel of this program was programmed in ASK. The ratio is thereby improved to 1:220 as the result of storing constants in ADB's, making more efficient address calculations, and increasing the amount of overlap between PE and PEM. Detailed analysis of the kernel of Glypnir and partially ASK-coded programs have been performed.

In solving Poisson's equation, the efficiencies of three popular iterative methods (SOR, SLOR and ADI) have been compared on ILLIAC IV. The storage schemes and methods of implementation which maximize the efficiency of these algorithms on ILLIAC IV have been investigated. The implementation of the direct method in solving Poisson's equations on rectangular regions with various boundary conditions is under way. Detailed reports of this work will be available next quarter. These studies on the efficiency of ILLIAC IV in atmospheric dynamics calculations are jointly supported by ARPA and the National Science Foundation.

(g) Algorithms for the "identification" of non-linear differential equations have been developed. Such algorithms will allow determination of certain sets of differential equations that satisfy a set of data provided as input. These routines have been implemented on the B6700 in Algol; however, many features are suitable for parallel machines. A document will soon be published. We believe that the developed package will be of interest to many of the ARPA community who are involved in mathematical model building. If the response is favorable, work will proceed on implementing the package on ILLIAC IV. This

is the first of three studies begun at the Center for mathematical model building. The other two studies are stochastic model building and identification of linear and periodic dynamic systems which are described in Section 5.1.5.

5.1.5 Time-Series Analysis

(a) A Fast Fourier Transform (FFT) subroutine has been written in ASK, and fully debugged and tested. Timing simulation results have shown that this program will perform Fourier transforms on ILLIAC IV with good efficiency. A descriptive document and a user's manual are available [21].

(b) Glypnir codes for a one-dimensional FFT, autocorrelation, and computation of the impulse response of a band-pass filter are fully debugged and tested. A document is forthcoming.

(c) A document has been completed [22] describing a B6700 package developed to determine the stochastic model used to describe observed data (which is in the form of time series), determine the order of the model, and estimate the parameters using a maximum likelihood. The feasibility of using parallel computations in this study, especially for the estimation algorithm, is being considered.

(d) Algorithms have been developed for the identification of linear time-invariant and periodic (period is not known a priori) dynamic systems using only the observed data (no control inputs). A B6700 Algol package is now being completed. The obtained results are very satisfactory. Many parts of the developed algorithms are suitable for a parallel computer, especially since the identification of a periodic system is reduced to the identification of many (64) linear system. Again, we believe this study to be valuable for researchers involved in mathematical model building (Economics, Biology, Engineering, etc.).

5.1.6 QUASCO: An ILLIAC IV Quadratic Assignment Code

As part of the ILLIAC IV algorithm development effort (and in support of CAC hardware design research) a modest effort is under way

to formulate and code efficient parallel procedures for treating the quadratic assignment problem. The results of these investigations to date, though incomplete, seem encouraging.

The quadratic assignment problem requires a one-to-one matching of the elements of two sets such that a quadratic objective function is optimized. For example, in the area of computer hardware design it is often desirable to assign discreet positions on a wiring board to some number of interconnected components in such a manner that the total length of connections between components is minimal. Other applications of the model relate to optimal industrial plant layout, hospital design, office space allocation, urban planning, etc.

A possible notation for the problem (in simplified form) is:

$$\min_{\rho} \sum_{i=1}^n \sum_{j=1}^n c_{\rho(i), \rho(j)} \cdot d_{i,j}$$

where:

$[C_{\rho(i), \rho(j)}]$ = an $n \times n$ matrix expressing the cost per unit distance between any two elements of the first set (components, plant functions, departments, offices, activities, etc.).

$[D_{i,j}]$ = an $n \times n$ matrix expressing the effective distance between any two elements of the second set (positions on a printed circuit board, spaces within a plant, hospital or office tower, city blocks, etc.).

ρ = a permutation of indices of the elements of the first set corresponding to a one-to-one matching to the elements of the second set.

While integer programming formulations of the problem are possible, the effect of the resulting quadratic objective function and integer constraints is always to create non-convexities in the solution space and hence numerous locally-optimal solutions. It may indeed be impossible to compute global optima for large quadratic assignment problems, even on ILLIAC IV.

Thus, our attack has focused more on the development of parallel procedures for approximating optimal solutions. Since these methods employ hill-climbing programming strategies, the solutions depend to some extent on starting points (initial feasible assignments derived by chance or design). Given such algorithms and an appropriate sampling strategy, however, the speed at which ILLIAC IV will be able to generate and rank alternative approximations will afford more confidence in the calculation and employment of best approximations to the optimal solution.

An existing quadratic assignment code developed by a staff member was re-written in Algol for the B6700 in order to realize efficient computer usage and inexpensive costs. Extensive improvements were made to the algorithm; however, it still remains in a state of dynamic change. Test results thus far indicate that QUASCO has achieved better results than other known algorithms. Work will proceed on an ILLIAC IV routine when this heuristic algorithm becomes better formulated.

A report is forthcoming in which are presented several ideas on the quadratic assignment problems and the approximation of one matrix by another of lower rank.

5.1.7 Graphics Algorithm Development for ILLIAC IV

5.1.7.1 Surface Presentation of a Function $F(X,Y)$

A basic ASK routine was completed for the perspective presentation of a graphical image of a surface represented by the function of two variables, X and Y . The design and implementation of this algorithm, however, is based on the still not rigorously defined input/output and disk access aspects of the ILLIAC IV operating system. Further work in sending the graphics data stream of this algorithm from ILLIAC IV into its control system has been suspended pending the suitable definition of the control system by Ames. As soon as the new I/O system to ILLIAC IV is defined, and the programming conventions established, this algorithm will be remolded to fit the new concepts.

5.1.7.2 Contour Plotting Algorithm for ILLIAC IV

A new contour plotting algorithm for the $M \times N$ array case, using real variables, was developed using SSK. The new implementation is a single program specifically designed to take advantage of parallel processing, both in the calculation of the contour data and in the resulting production of graphical images.

The current version of the algorithm assumes the case of an $M \times N$ array of data which is entirely core contained. Further work on this algorithm development was halted pending complete definition of the input/output system by Ames.

When work resumes, the case of an $M \times N$ array not core contained (partially or wholly on disk) will be completed and the input/output portions of the algorithm adapted to the new conventions for the Ames system.

5.1.8 Miscellaneous

(a) Two algorithms for finding the roots of polynomials with real coefficients have been programmed in Algol-Root-Squaring and an algorithm for finding the real roots using the Sturm sequence property. A parallel algorithm is now being designed to find the real roots of real polynomials using the Sturm sequence property.

(b) A short experiment in 8-fold precision on modular arithmetic was carried out using eight prime numbers furnished by the DeKalb Foundation for Number Theory.

(c) The block triangular decomposition of large matrices for eigenvalue problems by means of the transformation of $B \rightarrow PBP^{-1}$, where P is a permutation matrix, has been considered. The method used finds the transitive closure of the digraph G associated with B .

An Algol program was written and tested to put a 48×48 matrix in block triangular form, making use of the B6700's ability to logically find the "and" of two 48-bit words in one instruction. Then the program was extended to $n \times n$ matrices using the same method. The program works, and is now being documented.

The central part of the above program is the calculation of the transitive closure of a digraph. An ASK code to do this for a 64×64 matrix on ILLIAC IV using Warshall's algorithm [22] has been written; it will be extended to a $64n \times 64n$ matrix. The algorithm is extremely fast on ILLIAC IV compared to conventional machines.

(d) Other Algol routines added to the group's library were:

- Householder's reduction of a symmetric matrix to tri-diagonal form.
- Newton's method to obtain a root of $f(x) = 0$.
- Wegstein's secant method to obtain a root of $x = f(x)$.

The above secant method was programmed in ASK to run in a serial mode on ILLIAC IV.

5.2 NETWORK SYSTEMS AND SERVICES GROUP

5.2.1 Introduction

The Network Systems and Services Group is responsible for developing ARPA Network computer systems and services to serve the needs of the Center staff and user community. There are three project areas to be reported on at this time:

- (a) ARPA Network Terminal System (ANTS)
- (b) Center Graphics Support
- (c) Network Graphics Effort

5.2.2 ARPA Network Activities

5.2.2.1 ARPA Network Terminal System (ANTS) Development

Near the beginning of the reporting period, development of the ARPA Network Terminal System (ANTS) was seriously affected by failure of the B6700 performance to meet expectations. Progress was slow, erratic, but forward. Two full-time staff members have participated in the development of ANTS - one devoted to developing the high-level language compiler FEESPOL in which the system is written, while the other was responsible for the ANTS implementation.

On September 1, 1971, ANTS achieved Network operational status at the level of terminal access to the Network. Since September 1, efforts have been directed to consolidating the design and implementation of the full basic system in order to facilitate the system functions described below.

As of March 1, the basic system modules have been completed and the overall system structure set. Future efforts will be directed at implementing all applicable Network protocol operations, such as the low-level graphics protocol, data and file transfer protocols, extended terminal access protocols, including all required device drivers for peripheral devices, DECTapes, magnetic tapes, Gould printer, card reader,

disk and graphics displays. As of April 1, ANTS provides remote job entry access to the UCLA IBM 360/91.

5.2.2.2 ARPA Network Usage

With operational status of ANTS on September 1, 1971, the Center acquired minimal capability for connecting to the Network. It was quickly established, however, that the Network was in an early stage and service capabilities were undeveloped. Since September, experience on the Network has been primarily limited to attempts to use the BB&N service site in Cambridge and the Network Information Center at Stanford Research Institute.

More Network sites have become available during the period. With the recent addition of remote job entry capability to UCLA, Center usage of the Network will rise through the use of the GLYPLIT translator on the 360/91 and through the use of the number-crunching capabilities for running large codes. Additionally, the Center is aiding the University of California at San Diego in connecting their Burroughs B6700 to the Network by early June. This will enable us to transfer work now being done on our B6700 to San Diego via the Network. The Center's B6700 lease arrangement terminates on June 30, 1972.

5.2.2.3 Further Installations of ANTS Systems on the Network

During the Fall of 1971, the National Bureau of Standards (NBS) Information Processing Techniques group acquired and installed an ANTS system. Due to the hardware problems, both in the Network TIP and the NBS PDP-11 system, this site has not yet reached operational status. It is expected in the next quarter that installation of the site will be complete as well as installation of the proper drivers to handle the DCT 2000 integrated remote job entry terminal which NBS already has acquired and through which they intend to access the 360/91 at UCLA. NBS intends to use their version of ANTS to perform experiments and gather data on system performance, on the use of terminals to various ARPA Network sites, and on activities between systems on the Network. They will be modifying their operating version of ANTS through the PEESPOL

compiler on the UCSD B6700 to provide them with necessary input/output and data collection capabilities to facilitate these experiments.

Additionally, it is expected that two more installations of the ANTS systems will be made at Ft. Belvoir, Va. and Aberdeen, Md., both sites of the Army Material Command (AMC). Ft. Belvoir will use ANTS to interface remote job entry terminals to the Network and to connect a CDC 6600 to the Network. Aberdeen will obtain remote job entry access to the Network through ANTS to permit their AMC people to use the Ft. Belvoir CDC 6600 and other Network resources.

5.2.3 Graphics Support for Center Projects on the B6700

The development of a graphics package for the B6700 for use by the Center personnel was supported for two reasons: 1) there were a number of graphics devices on the PDP-11 which we wished to make available to people using the B6700; and 2) it was expected that this experience would aid in development of graphics packages for ILLIAC IV and any future Network systems in which we participated.

The connection of the PDP-11 to the B6700 was undertaken by writing a special operating system for the PDP-11 which attached it, via an 1800-baud telephone line, to the B6700 and made the PDP-11 peripherals look like terminals operating from the B6700 Datacom processor.

Further efforts were then directed at providing a basic graphics package with all rudimentary graphics operations such as line drawing, point drawing, lettering and scan graphics for faster scan conversion to the Gould electrostatic printer.

With the completion of this package in the Fall of 1971, efforts were then extended to the development of basic programming modules for plotting axes, scaling data, plotting lines from arrays, etc. In addition, a driving package was developed which allows one to interactively drive these programs to produce graphical imagery.

The decision was made to cease further development of B6700 graphics in January and to concentrate efforts on developing capabilities

to the Network and in the use of other Network host sites for Center personnel. Prior to this time, much experience was gained on the B6700 and some initial design on a generalized graphics data structure was completed and tested.

5.2.4 Network Graphics Efforts

5.2.4.1 Network Graphics Protocol

Several staff members are participating in the Network graphics group in an attempt to establish protocols for the transmission of graphical information across the Network. During the first quarter of 1971, agreement was reached on first level graphics protocol whereby graphical images were sent from the serving site to the using site in a very basic interpretive format allowing conversion into device specific format to such devices as the ARDS, Tektronix, Computek and IMLAC graphics display consoles. Future efforts will expand this interpretive language to provide for subroutining and dynamic interpretive display of subpictures.

5.2.4.2 Laboratory for Atmospheric Research (LAR) Support

With the advent of remote job entry to the UCLA 360/91, efforts are being directed at developing the necessary graphical routines and packages on the model 91 so that work there by Center and LAR personnel can be retransmitted back to the Center's graphical facilities attached to the PDP-11, i.e., the Computek display scope and the Gould electrostatic graphics printer. This effort will produce systems necessary for activities in the University's Atmospheric Research Laboratory under the direction of Professor Ogura. In addition to direct display from UCLA, a third party access through the Rand Tenex system will be performed whereby researchers interactively select their graphics displays and formats after their problem has been run and large output files produced.

5.3 ECONOMIC RESEARCH GROUP

During this period, the Economic Research Group has been engaged in an effort to develop a general manpower, economic and educational forecasting model which would be broadly useful to a large group of users in many agencies at the national, state and local level, to private businesses engaged in planning, and to scholars and students of economic, manpower and educational planning.

The Center's econometric research for the models used has been supported from several sources, including the U.S. Department of Labor, Manpower Administration and Offices of Policy Development, the University of Illinois, and ARPA. Further development of the econometric models will be supported by other agencies and foundations. ARPA support will be used to develop the user interface, ILLIAC IV computational system, and for development of a large applications system which is distributed over the Network.

The STEP I model, which is almost complete, forms the first version of this model. STEP II, which has begun, is a more complex version of STEP I entailing more sophisticated economic modelling and making much greater demands on computational systems. STEP I is implemented on conventional sequential computers. STEP II will require the computational capacity of ILLIAC IV because of the large computational capacity needed for matrix operations. The entire system by which remote users will access the model library data files, and computer utilities needed for economic, manpower and educational planning is termed MEASURE (Manpower and Economic Analysis System for Use in Research and Evaluation). MEASURE is to be developed in a form which is accessible to economists, planners, budget experts and others who are not trained in econometrics and computer science. Properly implemented, MEASURE can provide very large economies to users, and, ultimately, to the various governments whose needs create the demand for planning and forecasting. Most important, MEASURE permits the planner a much higher degree of freedom in

developing his plans since a much larger proportion of the analyst's time is devoted to specifying and evaluating his projections rather than in the mechanics of data collection, model specification, and data processing.

It is by providing the econometric skeleton, highly user-oriented computer interface, extensive data files, and efficient utilization of Network computers ranging from the PDP-10 through the IBM 360/91 to ILLIAC IV and the low-cost communications capability of the ARPA Network that MEASURE will substantially reduce the costs of any particular forecast and enlarge the community of planners and forecasters who can use sophisticated forecasting models. A detailed description of the STEP I and STEP II models can be found in CAC Document No. 32.

5.4 ILLIAC IV LANGUAGE DEVELOPMENT FOR THE PHASE II SYSTEM

5.4.1 Summary

A study of the need for a convenient linguistic and systemic means to specify and control timely flow of information between hierarchic memory levels in support of a non-core contained computation on ILLIAC IV was started this quarter. Incorporation of run time hardware allocation constraints into the data flow process is also a problem.

As an ideal solution, one would like to have a compiler which would accept the source program and the run-time storage allocations, and produce the appropriate set of ILLIAC IV, Memory Management Processor (MMP), PDP-10 and CCA datacomputer "programs" necessary to efficiently support, synchronize and perform the indicated computation. Work during the next six months will attempt to identify the framework and methodology necessary to support attainment of these goals.

5.5 ILLIAC IV INFORMATION RETRIEVAL AND STATISTICAL SYSTEM (IRSS)

5.5.1 Summary

An investigation was made and completed on information retrieval using ILLIAC IV. The conclusions reached were that the machine is not suited to the general problems of information retrieval and data management due to the awkward I/O structure, i.e., the lack of parallelism in I/O. However, files that can be linearized for serial searches are applicable to the ILLIAC since PE parallelism can be exploited. Serial searching is no longer a general problem of data management due to random access equipment and techniques. If serial search is necessary, it can be done on the IV; however, its use is probably limited.

The scope of the IRSS has been decided and includes the following:

- (a) The IRSS will relate to information retrieval, not to information management.
- (b) It will provide a user-language (non-procedural) and recognize data-item names.
- (c) The IRSS will provide data manipulating routines (statistical) identifiable through the user-language, operable in the ILLIAC IV.
- (d) The IRSS will be developed with the Ames Phase II system as its base.
- (e) The datacomputer will be used.
- (f) The IRSS will provide a means for researchers whose data reduction problems are large enough to warrant use of the ILLIAC IV to gain access to it.
- (g) Data transposition routines will be available for users who need a specific format of data in PE Memory.
- (h) The IRSS will be interactive but will not guarantee response in terms of completion.

With this emphasis, the next period will involve the development of the user-interface that is the non-procedural language. Based on the current state of the Ames I/O design, detailed design of the interface to the ILLIAC IV operating environment will be deferred. Design and implementation of the language, symbol tables, parser, and translation to data language will be investigated.

An initial version of a user-oriented interactive, file handling system called MONICA was completed during this reporting period. MONICA was developed to provide researchers easy access to a set of statistical subroutines. Emphasis was given to the user-interface and this experience will be used in developing the IRSS user-interface. A user's manual for MONICA will be produced in the next quarter.

Work was begun on defining user needs for an ILLIAC IV Statistical System and for reviewing existing systems. The ILLIAC Statistical System will be an integral part of the ILLIAC Information Retrieval System.

5.6 ILLIAC IV IMAGE PROCESSING SYSTEM

5.6.1 Summary

At the request of NASA's Earth Observations Program, the Center for Advanced Computation is exploring the application of the ILLIAC IV hardware system to the large-scale data storage and processing problems associated with the automatic interpretation and management of high-altitude aircraft and satellite earth resources imagery.

As background for this effort, the Center has surveyed existing methods of earth resources data collection by remote sensing, currently operational hardware and software systems for data processing, interpretation and information management, and meaningful applications of the technologies to specific problems of social concern. Preliminary investigations have shown the ILLIAC IV capable of efficiently processing, interpreting and managing large quantities of multi-spectral, digital imagery. The ARPA Network seems a practical means for interfacing the numerous, geographically-dispersed, potential users of such a system.

As a result of these initial determinations, the Center has joined with the Laboratory for the Applications of Remote Sensing of Purdue University, to design and implement a prototype ILLIAC IV-based information processing system.

5.7 EDUCATIONAL ACTIVITIES

5.7.1 Documentation

(a) Volume 1 of "An Introductory Description of the ILLIAC IV System" was published on July 15, 1971. Two hundred copies were sent to Ames and three hundred copies were kept at the University of Illinois. Following is an abstract of the book:

Written specifically for an applications programmer, the book presents a tutorial description of the ILLIAC IV System. Volume 1 contains three chapters-Background, Hardware Structure and the Assembly Language, ASK, as well as a Hardware Glossary. Many illustrative problems are used to educate the beginner in the use of the ILLIAC IV System.

(b) A very short "Ten Page Description of the ILLIAC IV System" is now in preparation. It was synopsized from "ILLIAC IV Hardware Structure," presented at the Ninth Annual Allerton Conference on Circuit and System Theory in October 1971 and from "The ILLIAC IV System," written for the IEEE.

5.7.2 Teaching and Consulting

(a) CS 491, "Architecture, Applications and Languages for a Parallel Computer" was offered at the University of Illinois as a graduate course in the Computer Science Department. The course has now been changed to CS 397 "The ILLIAC IV Computer System" and is being taught this semester.

(b) Short (one-hour to one-day), informal seminars dealing with the ILLIAC IV System were given to University of Illinois staff and students. Some consulting concerning ILLIAC IV was done by telephone with people outside the University community.

5.7.3 Training

Two Affirmative Action programmer trainees are being trained in the arts and crafts of programming for a digital computer and in algorithmic problem solving.

5.8 NARIS

The development of NARIS continued during this period. NARIS (Natural Resource Information System) is a geographic information system being developed in cooperation with the Northeast Illinois Resource Service Center. The Ford Foundation has provided funds for the initial research and development program.

The system is capable of supporting many remote users over telephone lines and providing conversational access to the geographic data base. A special purpose, user oriented language for retrieval and analysis of geographic information has been designed and implemented.

The system is currently in use at the Northeast Illinois Natural Resource Service Center at Lisle, Illinois, and the Northeastern Illinois Planning Commission (NIPC) in Chicago. NARIS runs on the CAC B6700 computer.

The data base contains a wide range of information on land use as well as information on geology, forestry, soil, and water characteristics of the selected area. Quarter-quarter section data (40 acre cells, one quarter mile square) on one township in each of the 8 counties is being inserted into the data base. It will take several more years to provide complete natural resource data at the quarter-quarter section resolution level for the entire 8 county area. The NIPC quarter section socio-economic data base is being broken down into quarter-quarter sections for input to the NARIS system.

Plans are under consideration to add a multi-resolution capability to NARIS so that it can handle more than quarter-quarter section data. For example, with the addition of multiple resolution capabilities, NARIS should be able to directly accept quarter section data from the NIPC files and statewide county resolution data to prepare a county data base.

5.9 IRIS

In May, 1971, the Illinois Institute for Environmental Quality contracted with the Center for Advanced Computation for a study of the need for and the feasibility of developing a geographic information system, the Illinois Resource Information System (IRIS).

The study examined the availability of geographically referenced data, the uses of such data in Illinois, and how State agencies produce and exchange such information. The state of the art in geographic information systems and the experiences of other states with such systems were also investigated.

In the course of the study, staff members of the Center for Advanced Computation conducted extensive library research and interviewed public officials in almost every department of State government and in several federal and local agencies.

During the feasibility study, three basic problems with the present system of geographic data collection, processing, and use were examined. First, Illinois suffers from a shortage of important geographic data -- data like current land uses and other fundamental data that are needed to make better decisions and plans. Second, there are no adequate procedures for communicating between agencies what data is needed and what data is available. Finally, the State needs more powerful and flexible tools to analyze the available geographic data.

The Center for Advanced Computation recommended that a computer based geographic information system, IRIS, be developed for Illinois to provide a common library of geographic data and capabilities for analyzing that data. Basic attributes of this system have been proposed with a detailed design to follow shortly. The Center also recommended that procedures be established for communicating data needs and data availability among State agencies. Final reports of the IRIS feasibility study will be submitted to the Institute for Environmental Quality early next quarter.

REFERENCES

1. McDaniel, L., ed., "Algorithm Development Group; Subroutine Library," Center for Advanced Computation, April 10, 1972.
2. Wilkinson, J. and Reinsch, C., Handbook of Automatic Computation, Vol. 2, Linear Algebra, Springer Verlag, 1971.
3. Wilkinson, J., The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1965.
4. Noh, K., "A Numerical Solution of the Matrix Riccati Equation," Center for Advanced Computation Document #24, 1972.
5. Bernhard, W., "ILLIAC IV Codes for Jacobi and Jacobi-like Algorithms," Center for Advanced Computation Document #19, 1971.
6. Sameh, A., "On Jacobi and Jacobi-like Algorithms for a Parallel Computer," Math. of Comp., July 1971, Vol. 25, No. 115, pp. 579-590.
7. McDaniel, L., "A Jacobi Algorithm for the ILLIAC IV," Center for Advanced Computation Document #21, 1971.
8. Sameh, A., "ILLIAC IV Applications," Proceedings of the Ninth Annual Conference on Circuit and System Theory," Oct. 608, 1971, pp. 1030-1038.
9. Bradbury, W., and Fletcher, R., "New Iterative Methods for Solutions of the Eigenproblem," Num. Math. 9, 1966, pp. 259-267.
10. Ogura, M., "A Parallel Scheme for Reducing a Real Matrix to the Upper-Hessenberg Form," Center for Advanced Computation Document #11, 1971.
11. Ogura, M., "The QR-Algorithm," Center for Advanced Computation Document #12, 1971.
12. Varah, J., "Rigorous machine bounds for the eigensystem of a general complex matrix," Math. of Comp., Vol. 22, 1968, pp. 793-801.
13. Varga, R., "On smallest isolated Gerschgorin disks for eigenvalues," Num. Math 6, 1964, pp. 366-376.
14. Medley, H. and Varga, R., "On smallest isolated Gerschgorin disks for eigenvalues, II," Num. Math 11, 1968, pp. 320-323.
15. Johnston, R., and Smith, B., "Calculations of best isolated Gerschgorin discs," Num. Math. 16, 1970, pp. 22-31.

16. Medley, H., "A note on a paper of Johnston and Smith's concerning best isolated Gerschgorin disks," Num. Math. 16, 1971, pp. 435-441.
17. Bernhard, W., "A general ordinary differential equation solver, GODES," Center for Advanced Computation Document #501, 1971.
18. Widlund, O., "On the use of fast methods for separable finite difference equations for general elliptic problems," Proceedings of Symposium on Sparse Matrices and Their Applications, Sept. 9-10, 1971, IBM Thomas J. Watson Research Center, Yorktown, N.Y.
19. Holt, H., and Masson, B., "The calculation of high subsonic flow past bodies by the method of integral relations," Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics, Berkeley, Springer-Verlag, N.Y., 1971.
20. Rajan, S., "Numerical solution of the partial differential equations of gas dynamics," Center for Advanced Computation Document #20, 1971.
21. Stevens, J., "A fast fourier transform subroutine for ILLIAC IV," Center for Advanced Computation Document #17, 1971.
22. Miura, K., "An introduction to the analysis of time series," Center for Advanced Computation Document #26, 1971.
23. Warshall, S., "A Theorem on Boolean Matrices, JACM 9, 1962, pp. 11-12.

DOCUMENTS AND PUBLICATIONS

- L. C. Abel, "Analysis of Simultaneous Operations and Memory Conflict in a Multimemory Computer System," CAC Document No. 5, University of Illinois at Urbana-Champaign, March 15, 1971.
- P. A. Alsberg, "OSL/2, An Operating System Language," CAC Document No. 6 (DCS Report No. 460), University of Illinois at Urbana-Champaign, June 10, 1971.
- P. A. Alsberg, "The Extensible Features of the Operating System Language OSL/2," Proceedings of the Third Symposium on Operating System Principles, Palo Alto, October 1971.
- W. H. Bernhard, "Godes/Godes: A General Ordinary Differential Equation Solver," CAC Document No. 501, University of Illinois at Urbana-Champaign, December 1, 1971.

- R. H. Bezdek, "Economics Research Group Working Paper No. 1, Progress Report on the Development of a Large Scale Conditional Consistent Economic and Manpower Forecasting Model," CAC Document No. 7, University of Illinois at Urbana-Champaign, July 27, 1971.
- R. H. Bezdek, "Economics Research Group Working Paper No. 2, Occupational Manpower Impacts of Shifting National Priorities," CAC Document No. 8, University of Illinois at Urbana-Champaign, July 27, 1971.
- R. H. Bezdek, "Economic Research Group Working Paper No. 3, Long-Range Planning in the Face of Variable Demands for Educated Manpower," CAC Document No. 13, University of Illinois at Urbana-Champaign, September 1, 1971.
- R. H. Bezdek, "Economic Research Group Working Paper No. 4, Manpower Analysis Within an Interdisciplinary Framework: Theoretical Potential and Empirical Problems," CAC Document No. 14, University of Illinois at Urbana-Champaign, September 1, 1971.
- R. H. Bezdek, et. al., "Economic Research Group Working Paper No. 5, The CAC Economic and Manpower Forecasting Model: Documentation and User's Guide," CAC Document No. 15, University of Illinois at Urbana-Champaign, October 15, 1971.
- W. J. Bouknight, "ILLIAC IV and the ARPA Network," Proceedings of the Allerton Conference, December 1971.
- W. J. Bouknight, et. al., "The ILLIAC IV System," Proceedings of the IEEE, to be published April 1972.
- C. K. Chen, "Solving Large and Sparse Zero-One Programming Problems on Parallel Processors," Ph.D. Thesis directed by D. L. Slotnick, University of Illinois at Urbana-Champaign, January 1972.
- S. A. Denenberg, "ILLIAC IV Hardware Structure," Proceedings of the Allerton Conference, December 1971.
- S. A. Denenberg, "An Introductory Description of the ILLIAC IV System," CAC Document No. 225, University of Illinois at Urbana-Champaign, July 15, 1971.
- A. L. Gaffney, "The Job Partner: An ILLIAC IV-B6500 Communications Monitor," M.S. Thesis directed by D. L. Slotnick, University of Illinois at Urbana-Champaign, January 1972.
- J. A. Garber, "Some ILLIAC IV Algorithms for Signal Processing," CAC Document No. 29, University of Illinois at Urbana-Champaign, March 1, 1972.

- D. M. Grothe, "PEESPOL Manual," CAC Document No. 500, University of Illinois at Urbana-Champaign, September 1, 1971.
- W. L. Heimerdinger, "Algorithms for the Identification of the Parameters of Linear Dynamic Systems," Ph.D. Thesis directed by D. L. Slotnick, January 1972.
- D. J. Kuck and A. Sameh, "Parallel Computation of Eigenvalues of Real Matrices," CAC Document No. 9, University of Illinois at Urbana-Champaign, November 1, 1971.
- J. E. LaFrance, "Syntax-Directed Error Recovery for Compilers," ILLIAC IV Document No. 249 (DCS Report No. 459), University of Illinois at Urbana-Champaign, June 21, 1971.
- G. LaVine, "Applications of Computer Technology to the Working Drawings Phase of Architecture," CAC Document No. 28, University of Illinois at Urbana-Champaign, March 1, 1972.
- N. C. Machado, "An Array Processor with a Large Number of Processing Elements," CAC Document No. 25, University of Illinois at Urbana-Champaign, January 1, 1972.
- I. W. Marceau and T. W. Mason, "A Matrix-Generator System for Linear Programming Problems," CAC Document No. 4, University of Illinois at Urbana-Champaign, June 30, 1971.
- K. Miura, "Parallel Radiation Transport Code: A Monte Carlo Method Applied to ILLIAC IV and its Statistical Consideration," CAC Document No. 18, University of Illinois at Urbana-Champaign, October 1, 1971.
- K. Miura, "An Introduction to the Analysis of Time Series," CAC Document No. 26, University of Illinois at Urbana-Champaign, February 22, 1972.
- K. Noh, "A Numerical Solution of the Matrix Riccati Equations," CAC Document No. 24, University of Illinois at Urbana-Champaign, January 20, 1972.
- M. Ogura, "A Parallel Scheme for Reducing a Real Matrix to the Upper Hessenberg Form," CAC Document No. 11, University of Illinois at Urbana-Champaign, September 1, 1971.
- M. Ogura, "The QR Algorithm," CAC Document No. 12, University of Illinois at Urbana-Champaign, September 1, 1971.
- S. Rajan, "Numerical Solution of the Partial Differential Equations of Gas Dynamics," CAC Document No. 20, University of Illinois at Urbana-Champaign, November 5, 1971.

- D. L. Slotnick, "Background and History of Parallel Processing," Proceedings of the Allerton Conference, December 1971.
- D. L. Slotnick, "Structure et Conception des Ordinateurs" (Architecture and Design of Digital Computers), Ecole d'Ete de l'O.T.A.N. sous la direction de Guy G. Boulaye, A Chapter, Computer Structures, Dunod, Paris 1972, pp. 19-41.
- B. Wittman, "An Investigation into Information Retrieval Utilizing ILLIAC IV," CAC Document No. 22, University of Illinois at Urbana-Champaign, November 30, 1971.

6. THEORY OF DIGITAL COMPUTER ARITHMETIC

(Supported in part by the National Science Foundation under Grant No. US NSF GJ-813.)

During the quarter, papers were prepared for presentation at the Symposium on Computer Arithmetic at College Park, Maryland on May 15 and 16, 1972. Although the format for the talks differs from the papers incorporated in the proceedings of the symposium, the talks collectively summarize the papers' contents. The papers are:

- 1) The Status of Investigations into the Use of
Continued Fractions for Computer Hardware
by James E. Robertson and Kishor Trivedi
- 2) Radix 16 Evaluation of Some Elementary Functions
by Milos D. Ercegovac
- 3) Application of Continued Fractions for Fast
Evaluation of Certain Functions on a Digital Computer
by Amnon Bracha.

These papers are to be submitted for publication in a special issue of the Transactions on Computers of the IEEE.

(James E. Robertson)

7. SWITCHING THEORY AND LOGICAL DESIGN

(Supported in part by the National Science Foundation under Grant Number U.S. NSF-GJ-503.)

Improvement in transformations for simplifying NOR networks was continued. Optimal networks with AND and OR gates were computed, assuming the availability of complemented inputs.

A computer program to design MOS networks with minimal numbers of MOS cells was completed by T. Shinozaki. Analysis of his computation with this program was completed as his master degree thesis. He pinpointed which part of the algorithm was most time-consuming. T.K. Liu worked on an algorithm to design an optimal feed-next network with MOS cells. This type of network is not only fast in processing a sequence of input data but also good in improving the packing density of IC implementation.

S. MUROGA

Transformation procedures for the simplifications of NOR networks were studied. The main results are as follows.

- (1) Modifications of calculation procedure of CSPP's (compatible sets of permissible functions). Since CSPP's are used in almost all our transformation procedures, a better procedure for CSPP's may make these transformations more powerful. We made several modifications for this purpose. Some of the modifications also help to reduce the calculation time.
- (2) Simplification of the transformation procedure which uses the concept of error compensation. This transformation may be the most powerful one among all of our procedures. One problem with the original version was the complexity. Our major efforts were devoted to simplifying the procedure

without losing much of the transformation power. The procedure finally obtained looks reasonably powerful and much simpler than the original version. Several examples have been tested by hand but further modifications may be required.

(3) Effective combination of simple transformations without using CSPF's were also studied.

(Y. Kambayashi)

Optimal AND-OR gate networks (both external variables and their components were available as inputs) for most functions of 4 or fewer variables were found. These networks are optimal in the sense that first the minimum required number of gates was determined, and then the number of connections was minimized.

There are 65,536 functions of 4 or fewer variables. These can be partitioned into 222 NPN-equivalence classes for the case when networks are to be realized by AND and OR gates with both complemented and uncomplemented external variables available. Of these 222 functions, we have obtained optimal networks for 212. And, of the remaining 10, 1 is known to require at most 9 gates, 2 require at most 8 gates, and 7 require at most 7 gates. The 7 gate problems (functions) which have been solved each required about 30-35 minutes of computation time.

Besides the AND-OR network problems, further improvements and modifications of NOR network transformation algorithms and programs were made. Related transformations were mentioned in earlier Quarterly Reports. Several different modified versions of a "pruning" transformation procedure, PROCIP (for PROCedure I Prime), were made and compared as to effectiveness and

speed. We were able to improve the effectiveness of the procedure for some examples without losing effectiveness for other examples.

(J.N. Culliney)

The first draft of the users manual for the all-interconnection program was completed and revisions are currently being made. The program was used to obtain optimal one-bit full adder networks with the complement of carry available as one of the outputs. Optimal NOR, NOR-AND, NOR-NAND, and AND-OR adder networks were obtained.

(K. Hohulin)

Further improvements on redundancy check procedures for NOR networks were done during this period. Several new versions of the program, which is aimed at obtaining optimal networks from a "universal network" (See Quarterly Report Oct.-Nov.-Dec., 1972), were developed and tested. One of the versions obtained an absolutely minimum NOR network for a 1-bit adder, which was not obtained by the previous version. This version, however, is not powerful enough to obtain absolute minimum NOR networks for all 3- variable single output functions. A somewhat more powerful procedure for these 3-variable functions was found, but the effort to develop new procedures is still necessary.

A procedure for merging two gates in a NOR network was coded, and some work to develop a program which produces a NOR network for a given set of output functions by a branch-and-bound method was done.

(H.C. Lai)

8. OFFICE OF THE SOUPAC CONSULTANTS

The fifth edition of the Soupac manual came out in February. The following programs made their appearance for the first time:

- Regression-Correlation Package
- Orthogonal Procrustes
- Scatter Plots
- Varisim
- Tucker-Messick Points of View Analysis
- Exact Restricted Least Squares
- Stochastic Restricted Least Squares
- Transportation Problem
- Crospa-Cross Spectral Analysis

In addition some writeups received major revisions:

- Introduction to Soupac
- T-Test
- Canonical Analysis
- Balanova
- Classification
- K-Class
- Autocorrelations
- Random Number Generator

Some of these writeup changes reflect substantial revisions in the programs.

Several new programs are being developed or converted for use in

Soupac:

- Calcomp Two-way Plot
- Polynomial Regression
- Personal Probability Space
- (new) Spectral Analysis
- (new) Linear Programming

The Soupac staff sponsored its series of Soupac Seminars to describe the use of Soupac to interested users. As many as 40 users attended the weekly sessions.

Soupac has been set up in the past at several locations outside of the University of Illinois. The following, therefore, received updates of the Soupac programs:

Cleveland State University
Washington, D.C., Bureau of Labor Statistics
University of Ottawa
State Farm in Bloomington, Illinois

The Soupac consultants have continued to give their assistance whenever called upon to classes and other groups interested in discussions of Soupac. Talks on Econometrics and Analysis of Variance were delivered to classes during the current period.

(Kern W. Dickman)

(Supported in part by the National Science Foundation under Grant Number US NSF GJ 27446)

The following is a collection of related work aimed at improved designs for computer and software systems. We are interested in parallel and pipeline processors, small primary memories, effective use of rotating memories, and some questions concerning user languages for problems including typical FORTRAN type calculations, simulation languages, and a variety of file processing problems.

(D. Kuck)

9.1. FORTRAN Parallelism Detection

A number of changes were made to the analyzer during this quarter, and some preliminary results have been obtained [1].

The Analyzer

The automatic FORTRAN analyzer program has been improved in several ways. The analyzer now accounts for operand fetches in both the serial and parallel processor cases. This will allow us to determine the effects of memory operation on overall parallel speedup.

A new weighted tree height reduction algorithm has been completed and will be integrated with the analyzer during the next quarter. This new algorithm allows us to account for weighted operators, i.e. operators requiring different amounts of execution time.

The scanner now produces various listings which are an aid in translating serial FORTRAN codes into parallel codes. These listings include array cooccurrence tables, array index patterns, and indications of parallelism within DO loops.

Finally, the presentation of statistical measures is being improved, a number of bugs have been eliminated, and we are continuing to investigate ways of improving the potential speedup through improved algorithms.

9.2. Preliminary Results

The following is an accounting of all the programs we have analyzed. In a few cases minor adjustments were made by hand, but essentially all of the following results were generated by our analyzer. No source deck was thrown out because it gave poor results. (See [1] for a more detailed discussion of our results.

Table 1 shows our numerical results; column heading explanations follow:

- 1) The names were derived from the original program's name. Note that GQU6Z, SQUANK and TRAUB were cut into two pieces to facilitate analysis.
- 2) This is the approximate number of source cards, ignoring comments. % DO is a count of all cards in the scopes of DO loops. Note that the total number of cards analyzed is nearly 1000.
- 3) This is the maximum number of DO iterations which we assumed for any loop in the deck for the measurement.
- 4) The number of traces is the sum of all paths from all entries to all END, STOP, or RETURN statements plus the number of IF loops. Several programs are broken into two sets because of a bimodal distribution of measured values. The predominance of IF and GO TO statements is implied by large values.
- 5,6) T_1 and T_p are computed as discussed earlier. In some cases we show a range of values; in others (with small variance) we show the mean. The simplicity of some of these programs is indicated by the small T_1 values.
- 7) This is the ratio of columns 5) and 6).
- 8) This is the number of processors required to achieve the T_p value of 6). In some cases we show p_{\max} . In other cases, the procedure of Section VI was used to increase the efficiency and the resulting value of p is shown.

9,10) The efficiency and utilization values are given for the number of processors of 8). When $E_p \approx U_p$ few redundant operations were introduced by our analyzer. Note that R_p was usually less than 2 or 3 and rarely over 5.

As we discussed earlier these values can be improved in several ways. We feel that by improving the analyzer most of the efficiencies below .25 could be doubled for the number of processors shown in the table. Furthermore, by decreasing the number of processors it is possible to increase the efficiency further. For example, the efficiencies of SQUANK2 a) could be improved by reducing p to 16. In this and a few other cases, the present state of our analyzer prevented us from carrying out the procedure of Section VI as mentioned in 8), above. It should be noted that some of the programs measured are sub-routines of others. Thus we could also achieve better overall statistics by averaging over traces that include several routines.

The major conclusion we draw from this is that a computer capable of executing 16 arithmetic operations simultaneously could be used effectively on an ordinary mix of small numerical FORTRAN programs. We use the word "effectively" advisedly. It seems that for almost all of the traces shown we could reorganize the programs to achieve E_{16} values with an overall mean in the range of .4 to .6. For a small residue of traces, $.1 \leq E_{16} \leq .2$ may be inevitable; e.g. for small IF loops and "default paths" through a program. However it should be recalled that our definition of E_p is a rather strict one. Even with present machines, "CPU utilization" figures are often discussed without regard to what the CPU is doing. Our U_p measures operations that are purposeful in that their introduction helps to speed up the program. It should be observed that the U_p values tabulated are all above .15 and that for some programs U_p is very much larger than E_p .

Table 1
Experimental Results

(1) Name	(2) Number of Source Cards (% DO, % other)	(3) Number of DO Itera- tions Assumed	(4) Number of Traces Measured	(5) T_1	(6) T_p	(7) S_p	(8) P	(9) E_p	(10) U_p
CHSTEP	45(0,100)	0	12	22-65	8-17	2.75-4.0	12-14	.23-.28	.62-.66
SQUANK1	80(0,100)	0	77	44-147	9-20	3-8.2	8-12	.32-.81	.62-.91
SQUANK2	60(0,100)	0	a) 63 b) 14	9-72 9-38	7-22 4-15	1.7-6.5 1.9-2.9	64 4-8	.026-.11 .29-.63	.21-.57 .34-.81
GQU6Z1	70(0,100)	0							
GQU6Z2	50(0,100)	0	a) 6 b) 14	12-40 30-94	4-15 9-26	1.2-4 2.9-4.2	3-5 16	.3-.8 .18-.26	.75-.85 .32-.54
FRANCZ	60(10,90)	10	a) 76 b) 30	355-444 17-55	74-106 3-23	4-5 2.4-6.7	20 7-12	.2-.25 .2-.83	.21-.25 .34-.96
CHVAR	35(20,80)	2	6	50	15	3.5	10	.22-.32	.4-.7
TRAUB1	60(35,65)	10	420	20-605	3-70	2.2-10.5	8-50	.07-.80	.35-.95
TRAUB2	40(30,70)	10	15	98-167	29-53	2.9-3.7	39	.07-.1	.24-.31
BOUNDZ	80(15,85)	10	a) 6 b) 29	11-26 98-257	9-15 10-59	1.2-1.7 2.8-12.5	2-6 28-30	.28-.66 .10-.41	.44-.72 .16-.41
EVALZ	8(25,75)	10	1	31	9	3.4	18	.19	.41
QSF	60(20,80)	22	a) 7 b) 1	16-59 633	6-18 82	2.6-4.3 7.7	3-20 44	.2-.88 .17	.45-.9 .30
FSERLZ	80(20,80)	16	3	549-580	32-38	15-17	64	.24-.26	
STUDENT	50(50,50)	3	5	195	58	3.3	17	.19	.25
AAAMA	40(60,40)	20	3	2700	110	25	40	.62	
ALG1	10(50,50)	20	1	176	11	16	60	.27	
ALG2	20(65,35)	7	2	216	42	5.1	28	.18	.24
ALG3	10(50,50)	20	1	216	13	16.6	40	.41	
SPLITZ	30(50,50)	30	1	365	43	8.5	60	.14	.24
ADIMPZ	60(15,85)	100	60	2700-2800	90-110	25	100	.25 ⁺	
DECMFZ	20(90,10)	20	1	5034	304	16.5	100	.16	.16

Ignoring questions of actual machine design, the speed-up measurements are of intrinsic interest. It may be observed that with the proper compiler and machine organization which reduce the data flow transients mentioned in Section II, but without changing the speed of components, the speed-ups shown (over a single processor) may be achieved for ordinary FORTRAN-like programs.

9.3. Simulation Processor - (E. W. Davis)

Parallelism within GPSS blocks has been measured for several frequently used block types. The original 360 assembly language routines were converted to FORTRAN for analysis on the system described in [1] and discussed in other sections of this report.

Results of analyzing six block types are indicating maximum speedups ranging from 4.2 to 6.0 for 8 to 12 processors with utilization near 50%. A total of 21 blocks have been converted and will be analyzed.

GPSS programs have been scanned by a PL/I program that marks blocks which are concurrently executable. Results, for several programs of various lengths, are that an average of two to three blocks can be processed simultaneously. This is a static analysis of the simulation program and does not reflect the possibility of more than one transaction being processed concurrently.

Total parallelism in a GPSS program is the product of the parallelism within a block, the parallelism between blocks, and the parallelism due to multiple transactions. The third factor is to be determined by a simulation of the concurrent execution of a GPSS program.

9.4. Text Searching

Basic Program - (W. H. Stellhorn)

Debugging via simulation is complete for the first version of the interactive language for text searching applications, and the necessary facilities

are now becoming available so that testing can begin on the D-machine itself. It is anticipated that this testing, with the possible exception of disk-handling routines, will be completed during the next quarter. In addition, techniques are being planned for measuring various aspects of system performance.

Two new features under development will allow a user (a) to operate on subsets of the data base which represent the response to previous inquiries and (b) to insert comments into the system to be stored with particular documents. These comments will remain separate from the text proper, but otherwise will be available for searching and printing in the same manner as the rest of the document.

Work is also underway toward analyzing the contents of our digitalized data base with the object of planning initial experiments in information retrieval.

9.5. S-Level Program - (J. R. Rinewalt)

The S-level program is being modified so that the text of each question asked by a user and a list of the documents which responded to each question can be saved. This will allow the user to list the text of any previous question and will allow him to restrict a search to the set of documents which responded to some logical combination of previous questions. At present, the entire data base is searched for each question. Flow charts for this modification are complete and some coding has been done.

9.6. Hardware Support - (L. A. Hollaar)

Major modifications to the D-machine were made to allow the loading

of the microstore under microprogram control. To the user, the change added a new memory instruction, micro-write. This change required the total reconstruction of the input section of the microstore, along with modifications to the timing circuits of the machine.

In addition to the micro-write, the ability to read the sixteen data switches on the front panel under program control, a means of selecting any micro-instruction for inspection or modification, and a real time clock readable under program control were added as part of the loader modification.

The card reader interface seems to be operating satisfactorily, although some difficulty seems to exist at high speeds. The printer interface has been constructed and is in the process of being installed and tested. The teletype interface has also been constructed, while a rough design for the disk interface exists. It should be noted that as the machine currently exists, only three devices can be connected to it at any one time. A modification can be made, however, to increase this number to seven.

Minor hardware malfunctions have also been found and corrected. In addition, the power distribution system has been replaced by heavier wire to lessen the power losses and increase reliability.

9.7. D-Machine Assembler - (E. Polley)

The assembler has been modified to produce object code for instructions of varying length. Previously all string and word instructions had been four words long (a word being 16 bits). Since many user programs relied heavily upon MOVE and JUMP instructions and because these instructions did not require all three available fields their length was reduced. This will significantly reduce core requirements for the text searching program.

The input-output instructions have been designed and coded into the assembler. It is anticipated that they will be operational by mid-May. This will make the assembler completely operational. A user's manual will be published sometime in June.

9.8. SPITBOL - (R. W. Zears)

The SPITBOL Data Base Processor for the D-machine is now debugged. It has a keyword grammar that allows text insertion, deletion, substitution, and transposition at three levels of operation.

The MICROPROGRAM for doing dumps of S-memory is flow-charted and mostly written, awaiting completion of other I/O routines. It may be called from either S-instruction or M-program, but returns control to the next S-instruction. It dumps S-memory in blocks of 16 words, Hex format. The S-instruction is variable length so that it may replace any instruction in the set.

- [1] Kuck, D. J., Muraoka, Y., and Chen, S. C., "On the Number of Operations Simultaneously Executable in FORTRAN-like Programs and Their Resulting Speed-Up," to be published in IEEE Trans. Computers.

10. COMPUTER SYSTEMS ANALYSIS

(Supported in part by the National Science Foundation under Grant No. NSF GJ 28289).

The goal of this research is the development of analytical tools for system modeling and analysis of real time computer networks. Priority assignment and job dispatching rules for a geographically distributed computer network are being investigated.

10.1 Computer Network Modeling. (W. Barr and L. Mills)

We are currently using a GPSS model to simulate the effects of queue length on waiting times within each priority class for nonpreemptive priority queues. A comparison of the data collected so far indicates a high degree of correlation between the theoretical results predicted by our analytical model and the observed statistics from the simulation.

Investigation has continued in our efforts to develop tools to enable network computers to achieve economic viability. As a first step, we have formulated an algorithm to allow the user to specify a series of deadlines for a job and associate with each deadline a relative reward. We then formulate, for each center, heuristic scheduling and priority assignment algorithms which guarantee that the deadlines of the highest priority jobs will be met, where possible. The algorithm further indicates those deadlines which are in danger of being missed. We then use our measure of cost effectiveness as the key to load-leveling within the network computer.

10.2 Center Throughput Analysis. (S. Mamrak and F. Salz)

A model of the CSO IBM 360/75 has been developed to use as a tool for evaluating system performance. Using a GPSS model as the simulation language,

numerous system statistics are being generated. At this point, they include execution time, wait time, turnaround time, CPU and core utilization, and queue sizes. These statistics, along with additional information provided by the simulation, enable us to evaluate current center operations and will facilitate testing recommended changes without their actual implementation into the operating system.

The multi-facted priority scheme presently in use to schedule jobs on the IBM 360/75 has been evaluated and we have proposed a new priority scheme. The new scheme assigns an initial static priority to a job based on its total OS time requirement. There are basically two priority classes determined by this initial assignment and jobs are ranked within their respective classes. This initial assignment is dynamically adjusted depending on 1) the ratio of I/O-bound to CPU-bound jobs in the system and 2) total system balance as measured by evaluating processor and memory usage.

We are also developing an expression for a job's total processing time which reflects the job's time under OS control as a function of waiting time for core requests, waiting time for I/O requests, and actual CPU seconds required for processing. This expression will be used in the GPSS simulation when the new priority scheme is evaluated.

Publications

Bowdon, Edward K., Sr., "Network Computer Analysis," Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R72-505, 1972.

ABSTRACT:

This research is aimed at developing analytical tools for system modeling and analysis of real time computer networks. We begin by formulating a geographically distributed network of computers.

From a postulation of the essential characteristics of our computer network, we develop a queueing theory model for a multiserver system with a finite length priority queue. Then, under the assumptions of Poisson input and exponentially distributed processing times, we utilize this idealized mathematical model to investigate the steady state stochastic behavior of jobs in the network. We are particularly interested in the efficiency of computer utilization and the average waiting time for jobs of different priority classes.

We conclude by employing the average waiting time results from the queueing theory model to determine steady state job dispatching rules for a network of centers. From a knowledge of the average arrival rates and processing times for each priority class, we determine what fraction of the jobs in each class should be transmitted between centers in order to balance the average waiting times for each priority class throughout the network.

A summary of this work appears in the following:

Bowdon, Edward K. Sr., "Network Computer Analysis," Proceedings of the Fifth Hawaii International Conference on System Sciences, January, 1972.

Bowdon, Edward K., Sr. and W. J. Barr, "Throughput Optimization in Network Computers," Proceedings of the Fifth Hawaii International Conference on System Sciences, January, 1972.

ABSTRACT:

In this paper we propose a new measure of goodness for determining computer system performance. After developing our measure, we relate its variables to the system control parameters which are under the designer's (user's) control. Then, we extend the technique to determine throughput values for an interconnected center of computers. Two classes of centers are discussed and the technique is shown applicable to any center. The extension is continued to provide a measure for a geographically distributed network of computers. The existence of the measure provides us with the capability to compare centers within a net and hence to quantitatively determine criteria for job dispatching between centers.

Bowdon, Edward K., Sr. and W. J. Barr, "Cost Effective Priority Assignment in Network Computers," Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R72-509, 1972.

ABSTRACT:

With the advent of network computers, a new area of computer systems analysis has evolved. Unfortunately, most of the work which has been done to date merely extends the previously existing theory of communications. While this work has been very fruitful and produced important results, our analysis is predicated on the assumption that a geographically distributed network computer is, in reality, quite different from telephone networks and individual computing centers.

In this paper, we focus our attention on the probable goals of the networks and define a measure of cost effectiveness. Then using this measure we develop a priority assignment technique for the individual centers that comprise the network. We conclude by expanding the measure of cost effectiveness to determine load leveling rules for the entire network.

Bowdon, Edward K., Sr., "Dispatching in Network Computers," Proceedings of the Symposium on Computer-Communications, Networks and Tele-traffic, April, 1972.

ABSTRACT:

We begin by formulating a geographically distributed network of computers, the major characteristics of which are that:

1. It is a geographically distributed network consisting of centers interconnected by common carrier links.
2. It is a symmetrical structure in which common data files are stored in secondary storage devices readily accessible by all computers in the network.
3. It is a controlled structure which contains a distributed control program that regulates the allocation of facilities and the assignment of work to the computers in the network.

These characteristics allow parallel processing among all computing facilities within the network without the need to preplan the exact distribution of work.

We presume that we have a command network, one computing center in the network having been designated as the command center. The command center periodically polls the other centers and depending on the queue levels, requests arrival rate and processing time statistics. The command center determines which centers should transmit jobs and what fraction of the jobs in each class should be transmitted, in order to balance the average waiting times for each priority class throughout the network. Even though the average waiting time by priority class is not a linear function of arrival rate, we show that the steady state equations for waiting times can be used on a successive approximation basis to perform dynamic load regulation in the network. As a result, the command center can effectively utilize the intercenter communications facilities to improve the overall network throughput.

(E. K. Bowdon)

11. NUMERICAL ANALYSIS

11.1. Extensions of Factorization Methods and Adaptive Algorithms

During this quarter, work continued on the numerical solution of the mass transport equation, as it arises in the modeling of the dispersion of liquid waste in porous substrata. Features of the numerical approximation which have been considered are as follows:

Two equations must be solved, one of which includes first order terms. The first order terms tend to dominate the second order terms and the time discretization may reflect this in novel ways. The work of K. H. Coats* may bear on this.

The space variables are approximated by a variational method. Factorization procedures are being designed to apply to the solution of the resulting linear systems. The numerical properties of these algorithms are being analyzed mathematically for applications to the iterative procedures. These algorithms are more complicated than for those resulting from finite difference approximations.

The associated iterative procedure is an adaptive algorithm, which estimates acceleration parameters automatically.** Modifications of this algorithm are being made for Neumann boundary conditions.

Paul E. Saylor

* K. D. Coats, Journal of Society of Petroleum Engineers, Volumes 6,...11.

**Diamond, Martin A., "An Economical Algorithm for the Solution of Elliptic Difference Equations Independent of User-Supplied Parameters," Department of Computer Science, Report No. UI-UCDCS-R-71-492, December 5, 1971.

11.2. Eigenvalue Bounds for the Iteration Matrix of Stone's Factorization

Experimental results have demonstrated the rapid convergence of factorization techniques for the solution of large sparse systems of linear algebraic equations. In particular Stone's symmetric factorization has proved to be superior to SOR and ADI for a wide variety of difficult problems [3]. Factorization techniques are applied to a system $Ax=q$ by (1) defining an auxiliary matrix B so that $A+B$ factors into sparse triangular matrices L and U and (2) computing the approximate solution from the iteration.

$$(A+B)x_{n+1} = (A+B)x_n - \tau_n (Ax_n - q) ,$$

where the parameters $\{\tau_n\}$ are chosen to assure convergence. For A and $A+B$ positive definite, the optimal sequence of parameters is a Tchebychev sequence derived from the extreme eigenvalues of $(A+B)^{-1}A$. (The extreme eigenvalues are the largest and smallest eigenvalues.) By using the adaptive algorithm developed by Diamond [2], Stone's factorization may be applied without a priori estimates of the eigenvalues of $(A+B)^{-1}A$ and still have guaranteed convergence at an optimal rate. The rate of convergence may be initially slower but will increase so that error reduction obtained is approximately equal to the error reduction resulting from the use of optimal parameters.

Rapid convergence depends upon the eigenvalues of $(A+B)^{-1}A$ which determine the parameters $\{\tau_n\}$. Thus the adaptive algorithm can provide convergence at an optimal rate, but the value of this rate depends upon the matrix B. By combining our previous results on the

limiting values of the elements of $A+B_s$ where B_s is Stone's symmetric factorization [2] with the eigenvalue bounds derived by Bracha [1] we have shown the following. If the nonzero diagonals of A are constant, then the largest eigenvalue of $(A+B_s)^{-1}A$ is bounded above independent of the order of the matrix A . We have also shown that if A is derived from the Laplace's equation then the smallest eigenvalue of $(A+B)^{-1}A$ is bounded below independent of the order of A .

We conjecture the existence of similar upper and lower bounds for the eigenvalues of $(A+B)^{-1}A$ when A is any matrix derived from a five point finite difference approximation of an elliptic partial differential equation and B is defined by Stone's symmetric factorization.

Martin Diamond

- [1] Bracha, A. "A Symmetric Procedure for the Solution of Elliptic Boundary Value Problems", Department of Computer Science, University of Illinois, Urbana, Illinois, Report No. 440, April, 1971.
- [2] Diamond, M.A., "An Economical Algorithm For the Solution of Finite Difference Equations," Department of Computer Science, University of Illinois, Urbana, Illinois, Report No. UIUCDCS-R-71-492, December, 1971.
- [3] Taranto, R., "Numerical Studies of Stone's Symmetric Factorization and the Iteration Parameters α and τ ," Department of Computer Science, University of Illinois, Urbana, Illinois, Report No. 423, January, 1971.

11.3. A Fast Direct Method for the Solution of Linear Equations

We have started work on a fast direct method of solving linear systems of algebraic equations $Ax=q$ where A is a sparse nonsingular $N \times N$ band matrix with band width m . Virtually all matrices which arise in the solution of partial differential equations are of this form. The method is not as fast as some of the recently developed direct methods which can be applied when additional requirements are placed on A . However, it is much faster than direct elimination. The method requires $O(Nm-m^2+m^3)$ arithmetic operations whereas Gaussian elimination requires $O(N^2)$ to $O(N^3)$ operations depending upon the structure of A . If $m \leq \sqrt{N}$, which is often the case, the method requires $O(N^{3/2})$ operations.

The basic idea in the method is to find m linearly independent solutions of $(N-m)$ of the equations and then solve an $m \times m$ dense system of equations to determine the proper linear combination, of the m solutions, which is a solution of all N equations.

We are currently studying a stability problem in the method. The method has worked well on systems of up to 300 equations. However, on systems of 1000 equations the relative error has been as large as 10^{35} .

Martin Diamond

12. NUMERICAL METHODS, COMPUTER ARITHMETIC AND ARTIFICIAL LANGUAGES

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 812.)

12.1 MIPS

Work is progressing on a new batch operating system for the PDP-11. A common deficiency of existing batch systems is that they waste nearly 50% of the available CPU time due to lack of overlap of processing with the primary I/O operations of reading cards and printing listings. The new system is intended to handle a quasi multi-programmed mix of six jobs, and provide near optimal throughput for the typical student job mix. The user will have access to a simple, fast, and flexible filing system and will execute his programs on a 32K Virtual PDP-11.

The system is intended to run in a 16K word PDP 11/20 but will use any additional core to increase speed. The supervisor is completely resident and occupies the lower 6K of core. The rest of available memory is broken up into a "FILE" partition 2K words in size, and an "EXEC" partition which occupies all remaining core. The primary I/O tasks of reading cards and printing listings are handled by resident device drivers in supervisor core. Filing system code, console TTY drivers, job scheduler, swapping routines, and disk and core management routines are all part of the supervisor. Non-resident I/O bound system tasks such as tape to disk transfer routines, etc. will run in "FILE" core. System routines running in "EXEC" core will include the assembler, loader, interpreter, and other large programs which communicate only with the disk. CPU time is allocated on a priority basis where device interrupt service routines run at priority

4 or greater, supervisor tasks (such as the spooling routines) run at priority 3, "FILE" tasks run at priority 2, and finally "EXEC" tasks run at the lowest priority of 1.

Just like in the big time, a user job will consist of source decks, data decks, binary object decks, and appropriate control cards. On a JOB card the user will claim a number of disk blocks. The system will worry about finding disk space to hold the job's pseudo reader and will delay execution of the job until the number of disk blocks claimed become available. Once in execution, the job will sequence through its various tasks running in FILE or EXEC core as required. A job task running in EXEC core will be time sliced to prevent long jobs from tying up core for extended periods. The time slice will be long (possibly as long as 30 seconds) since slicing is only intended to provide rapid turnaround for short jobs. Blocks of unnamed files (pseudo readers, listed output files, and files created by the system but not specifically named by the user) will be deleted as they are used in order to free disk space as soon as possible. When a job completes, its listed output will be printed and all disk space still assigned to the job will be deallocated.

Due to the large access time and very limited capacity of the disk (512K characters), the disk management philosophy was of primary importance in system design. The system views the disk essentially as a paging device. Core images of user object code, all files and even the system routines are broken up into 256 word pages or blocks and scattered randomly over the available 1024 disk blocks. No contiguous disk space is allocated or reserved for any purpose. Furthermore, disk blocks (start addresses on disk) are skewed to allow optimization of disk access time. On bulk transfers such as swapping of an assembler

into core, the blocks may be transferred in any random order prescribed by the disk optimization routines. Blocks of a pseudo reader file being spooled to disk, blocks of a listed output file going to core for printing, or any other block going to or from disk may be interspersed with blocks of the assembler being swapped into core. Although little quantitative data on disk traffic is available at this time, it appears that this disk organization and the associated optimization routines will have a very favorable impact on system performance.

Currently, the filing system, disk queueing and optimization routines, all important device drivers, swapping routines, and parts of the scheduler have been coded. The loader is finished and the interpreter is in the coding phase. Necessary modifications to a PAL-11A assembler have not been started. Completion of all coding and start of debugging is scheduled for about the 1st of June.

(Jim Miller)

12.2 GIZMO

12.2.1 Introduction

Work is presently being done to expand the capabilities of GIZMO, the CAI system being implemented on the PDP-11. The present version of GIZMO, completed a few months ago, is in working order and the new additions described below are well into the coding stage. GIZMO's main purpose is to interact with students in programming courses in a dynamic teaching environment.

12.2.2 What GIZMO is Now

Presently, GIZMO's basic teacher mode has been completed. This is the interactive part which permits teachers to write lessons easily and quickly without the teacher having to spend much time prior to sitting down at the console. Teachers input their answers in a BNF-like language as described in Appendix A. Also completed is the basic student mode which enables students to interact with the previously compiled lessons.

The teacher mode feeds instructions to the teacher so that it becomes almost impossible for him to make mistakes. If the teacher does do something wrong, teacher mode will inform him and ask for clarification. Upon termination of the lesson, the teacher mode outputs a lesson in source format to a disk file handled by ETS. The teacher has the option of obtaining hand copy of source (papertape) or simply compiling the lesson by the lesson compiler into an object lesson. (Lesson compiler not yet finished.) Now the lesson is in a form in which a student may access by simply identifying himself to the student mode and naming a lesson present on disk in object form.

A number of special options active during student mode can presently be specified during teacher mode by the teacher. These include evaluator, ignorer and scruncher (see Appendix A for explanation).

12.2.3 Near Future Plans

In the near future, we plan to implement a number of new routines. Included in this repertoire are completion of the lesson compiler, and overlapping some major segments of code by making some mutually exclusive parts non-resident. Also planned is an independent program, a lesson editor which is lesson-part oriented which could be used by a teacher to change, add, or delete any question, answer, or option specified. This would be facilitated with such commands as:

- (1) Proceed to question (answer) n
- (2) Skip n questions (answers)
- (3) Delete question (answer), etc.

These would make it possible for a teacher to edit a lesson without the necessity for the teacher to understand, or even see the internal structure of a lesson.

In addition to the above programs, we also wish to add additional special options. For example: Timer, which would be used to specify how much time the teacher wants the student to have before responding (good for exams); Keyword, which would be used to search the student's response for specific keywords and therefore, give more meaningful teacher responses; Random number generator, to enable GIZMO, under teacher-direction to supply students with different data on each access to a particular lesson; Help, which could be specified

by the teacher to permit students to type HELP and permit GIZMO to give such; GIVE-UP, which could be specified by the teacher to permit students to type GIVE-UP and permit GIZMO to give the student the correct answer.

12.2.4 Long-Range Goals

In the future, we would also like to implement the capability to permit teachers to define within the BNF-like language recursively defined terms. We also plan to add simulators as part of the options so that a student may write a program at the console after a request to do so and then have a simulator dynamically evaluate the program on the basis of speed, clarity, space-efficiency, etc. Also, since ETS supplies ETS-users with such a wide variety of services not required by GIZMO, the possibility of developing a stand-alone GIZMO with its own teletype-polling and filing system would be an economic advantage.

12.2.5 So what's so new about GIZMO?

In view of the types of problems for which GIZMO has been designed, it has a number of major advantages over existing CAI's:

- (1) The ease of lesson writing. Previous systems have required the teacher to learn an entirely new language before being able to write a lesson. GIZMO's author-language is a simple BNF-like language which requires little advanced knowledge other than BNF. The teacher's manual is available (see Appendix A) as a reference guide for the lesson writer. It explains in simple language what must be done by the teacher. Even without the manual, it is conceivable that a teacher could write an excellent lesson on his first attempt.

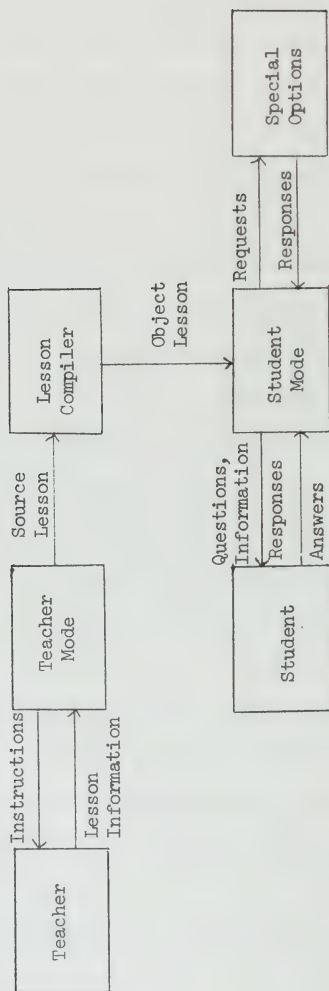
- (2) Students can specify either help or give-up which will both reduce frustration on the student's part.
- (3) GIZMO will initially be used in conjunction with teaching CS 201. Since these students are going to be learning PDP-11 assembly language (PAL), it is good policy that GIZMO itself is implemented on the PDP-11. First, it gives students a "feel" for the machine (i.e., interaction and realistic response time). Second, it enables the students to get on the machine on the very first day. Third, since GIZMO is well structured and strongly supported with both internal and external documentation, students may use GIZMO as a "model" large program.
- (4) Since the Eleven's CPU can be devoted to GIZMO and can perform with little effort (e.g., response time is effectively zero even during peak usage), the system can be designed to perform large scale simulations and dynamic examination of assembly and execution of student's programs.

12.2.6 Conclusions

GIZMO appears to have strong capabilities of effectively augmenting the traditional methods of teaching Basic Computer Science and assembly language courses. With a complete repertoire of lessons for CS 201 to be written this summer, GIZMO will be ready this fall, for its first major trial.

If any readers have any suggestions or would like to see a demonstration please contact me.

(Al Davis)



Note: All arrows signify data flow and are handled by Educational Timesharing System (ETS)

Figure 1 GIZMO Structure

12.3 Appendix A

12.3.1 Teacher's Manual for Teacher Mode of GIZMO

Writing a lesson for GIZMO is easy...and fun! Here are the simple step by step instructions. Underlined text is machine generated. The rest is user's.

A. Initial dialog with ETS.

.HE 2 7 DUMBO7
.SI 5
.RUN GIZMO

B. Initial dialog with GIZMO.

GIZMO VER. V-4 2/72
PLEASE IDENTIFY YOURSELF
GIZMO T*XXXX

where XXXX is the 4 character teacher code word which will be given to you upon receipt of your application.

LESSON NAME 1-6 CHAR

You should type a 1-6 character title which you would like to call your lesson. This is the same name that your students will use to identify the lesson.

C. Creating the lesson - Global states.

GIZMO will ask you a series of questions after each of which he will wait for your responses. GIZMO will check your answers for correct syntax and will give you the question

WHAT?

if your response is wrong. Whenever GIZMO says WHAT?, you should retype the last line correctly as it was not acceptable in its present state.

1. The first question will be:

GIVE FIELD DELIMITERS

This asks you to define what type of brackets you would like to use to define your BNF field names. To indicate default, type

↵

Default is < >. To indicate other types of brackets, type them followed by a ↵. These two brackets must be unequal, non-blank, and non-CR, LF. If these rules are not followed, GIZMO will type WHAT?. Please try again.

2. The next question will be:

GIVE AN OPTION

This asks you to indicate which of the available options you would like to have implemented while the student is taking your lesson.

The available options are:

a. Evaluator. This option specifies that all numeric expressions in the student's response will be evaluated as a PAL expression and then its value will be compared to the number in your indicated answer. For example, if you indicate MOV #3, R0 as a correct answer and evaluator is set, then MOV #5-10.+3+7, R2-2 would be graded as correct. To specify option evaluator, type:

E ↵

b. Ignorer. This option specifies that only up to a certain character in the student's response should be examined and that all characters typed afterwards will be ignored. For example, if you indicate that the correct answer is MOV #3, R0 and you have also indicated ignorer-semicolon, then

MOV #3, R0 ; comment

would be graded as correct. To specify option ignorer,
type:

IX ↵

where X is the character as defined above.

c. Scruncher. This option specifies that all occurrences of a certain character in the student's response should be skipped. For example, if you indicate the correct answer is MOV #3, RO and you have also indicated scruncher-blank, then

MOV #3 , RO

would be graded as correct. To specify option scruncher,
type:

SX ↵

where X is the character to be scrunched. However, if you specify a character in your answer that is also scrunchable, that character will be considered required in the student's answer to produce a correct match.

Note: For all the options above, you must give the correct number of parameters (0 for E, 1 for I, 1 for S) to indicate the character. You may indicate as many options as you wish. After each option entry, GIZMO will ask you the same question again. To terminate option entries, type

↵

If you specify the same character to be scrunched and ignored, the character will only be scrunched.

D. Creating the lesson - individual questions.

1. The first question will be:

QUESTION # AND QUESTION

This asks you to indicate the text which you would like to have printed to the student. If you wish to terminate the lesson type . (See part E.)

2. The next question will be:

HOW MANY TRIES?

This asks you to indicate how many times you want the student to have to guess wrong answers before giving him the correct answer or any other message that you wish. You may type as many characters here that you wish but only the first will be examined. It, therefore, must be an integer less than ten, and greater than zero. If not, GIZMO will type WHAT? and you should try again.

3. The next question will be:

GIVE A RIGHT ANSWER

This asks you to indicate an answer which should be graded as correct by GIZMO. It must be syntactically correct as defined in the section "BNF for right and wrong answers." If incorrect syntax exists, GIZMO will type WHAT? and you should check your response for the correct syntax as defined below. GIZMO will continue to ask for right answers until you type

4. The next question will be:

GIVE A WRONG ANSWER

This asks you to indicate an answer which should be graded as incorrect by GIZMO. It must be syntactically correct as defined

in the section "BNF for right and wrong answers." If incorrect syntax exists, GIZMO will type WHAT? and you should check your response for the correct syntax as defined below. GIZMO will continue to ask for wrong answers until you type

↵

5. The next question will be:

GIVE QUESTIONS MESSAGE AFTER X INCORRECT RESPONSES

This asks you to indicate what message you wish to have printed to the student if he fails to answer the question correctly after the number of times that you have indicated above. This MUST be a series of comment numbers correctly bracketed as defined below. If you attempt to insert field labels they will be accepted but nothing after the label's appearance will be printed to the student.

6. GIZMO will now ask you to define those labels and comments that you have used in the previous question and were not defined previously by you. Note, by the way, that all label defines and comment defines defined in this section will be global to the entire lesson. All labels must be defined in terms of "BNF for definitions" as described below. All comment defines will be accepted exactly as you type them.

E. Terminating the lesson.

When you type ↵ in response to a QUESTION # AND QUESTION, you will be asked to verify this action. In response to

QUIT! ARE SURE?

type YES ↵. At this time, your lessons (two files: SRC and COM) will be outputted onto the paper tape punch. Label these two paper tapes

with the lesson name, date and whether they are SRC or COM (the first one will be SRC, second COM).

The COM file is now ready for student's use but the SRC file must be compiled by GIZCOM to translate it into the OBJ file which then, in conjunction with COM, defines an entire lesson which is student usable.

F. BNF-like language for right and wrong answers.

Right and wrong answers can be specified as straight characters or BNF or any combination of the two. To use BNF here, you may use either label names or comment names--both must be prefixed and suffixed by the appropriate brackets that you defined above. Label names must have between 1 and 6 characters (A-Z only) inclusive. Comment names must have 1 to 3 digits (0-9 only) prefixed by an asterisk.

Examples of good label names are (using default bracket defines):

```
<ABC>
<A>
<WHA>
<MONKEY>
```

Examples of bad label names are:

```
< >          (too small)
<DONKIES>     (too large)
<AB123>       (no numbers allowed)
<LABEL 1>     (no numbers allowed)
< A >         (no blanks allowed)
```

Examples of good comment names are:

```
<*1>
<*35>
<*49>
```


Examples of bad comment names are:

<*>	(too small)
<*7393>	(too large)
<*1ABC>	(no letters allowed)

Examples of correct BNF for right and wrong answers:

```
MOV #1, RO
MOV <NUMBER>, <REGIST>
<ANSWER>
MOV <*1>#<*2>, <*3>RO<*4>
<OPCODE> <*1> <ONEAND> <*2> <COMMA> <*3> <TWOAND> <*4>
```

The indicated comments will be printed only if the student matches all of the label fields indicated before it on the line.

G. BNF-like language for label definitions.

The legal BNF symbols which you must know here are | and the brackets (as you defined or defaulted to < > above).

| or ↑ means "or"

Thus <AB|CD> means AB or CD. At this step you may not define new labels; all characters typed (except for BNF symbols, obviously) will be compared directly to the student's response. Thus, in the above example <AB|CD> will match either the characters AB or the characters CD, not the fields defined by those labels.

You may nest brackets 2 deep only if the inner one is a comment name. However, you may have as many of them as you want linearly. Thus, some examples of correct label defines are:

<A B>	(A or B)
<,>	(comma or nothing)
<>	(null - matches anything)
<., >	(period or comma or space or null)
<ABCD>	(the character string ABCD)
<A B> <,> <., > <ABCD>	
ABCD	(the character string ABCD)
MOV 	(the character string MOV B or MOV)
<A <*1> B <*2>>	(the character string A with comment 1 or the character string B with comment 2)

MOV <B <*3>|>

(the character string MOV_B with
comment 3 or the character string
MOV with no comment)

H. Example of a one question lesson:

.HE 2 7 DUMBO7,
.SI 5,
.RUN GIZMO,
GIZMO VER.VOL 2/72
PLEASE IDENTIFY YOURSELF
GIZMO T*0777
LESSON NAME 1-6 CHARACTERS
ADD1,
GIVE FIELD DELIMITERS
,,
GIVE AN OPTION
E,
GIVE AN OPTION
I;,
GIVE AN OPTION
S,
GIVE AN OPTION
STA_B,
GIVE AN OPTION
,
QUESTION # AND QUESTION
1. WRITE A PAL STATEMENT THAT ADDS 1 TO REGISTER 2,
HOW MANY TRIES? 3;
GIVE A RIGHT ANSWER
INC <REG> <*1>,
GIVE A RIGHT ANSWER
ADD #1, <REG> <*2>,
GIVE A RIGHT ANSWER
,
GIVE A WRONG ANSWER
<BADCOD> <BADNVM> <BADCOM> <BADREG>,
GIVE A WRONG ANSWER
<BADINC> <BADREG>,
GIVE A WRONG ANSWER
,,
GIVE QUESTION'S RESPONSE AFTER 3 INCORRECT ANSWERS
<*3>,
<REG>::= <%|R>2,
<*1>::= VERY GOOD,
<*2>::= OK. BUT INC R2 IS BETTER;
<BADCOD>::= ADD<B<*4A|>1,
<BADNVM>::= <#|<*5>1,
<BADREG>::= <2<*6>|REG2<*6>>1,
<BADINC>::= INC<B<*7>|>1
<*3>::= YOU'D BETTER STUDY MORE, BOY! THE CORRECT ANSWER
IS INC R2.

<*4>: := ILLEGAL OPCODE ↵
<*5>: := YOU NEED A # SIGN ↵
<*6>: := THAT'S NOT HOW YOU SPECIFY REGISTER 2 ↵
<*7>: := THE INCB WILL ONLY WORK IN SOME CASES ↵
QUESTION # AND QUESTION

↵
QUIT! ARE YOU SURE?
YES
TAPE OPERATIONS COMPLETE

12.4 PAL-11 Source Compression

Work continued on a program to compress PDP-11 assembler source decks so that more source code can be stored on disk. Compression is accomplished by: 1) replacing blanks between fields on the source card with a TAB character, 2) deleting the colon following each label, 3) deleting the semicolon appearing before each comment, 4) removing all trailing blanks, 5) replacing carriage return line feed by a single character, 6) replacing character pairs appearing more than once by single characters. Future versions will compress standard character pairs (e.g., RO, R1 SP, -(,)+, etc.) before searching the text for other pairs. The only storage needed is for the source text (which is overwritten with the compressed text) and for the compression table (128 word table containing the character pairs and their replacements). Two test PAL source decks were compressed. Deck #1, with 73 cards (5840 bytes), was compressed into 1077 bytes. The compressed deck was 18% of its original size. Deck #2 was compressed from 16640 bytes (208 cards) into 5141 bytes, 31% of its original size. (All numbers are base 10.)

(Barry Finkel)

Reference

B. A. Marron, "Automatic Data Compression," CACM 10, November 1967, pp. 711-715.

12.5 ETS

For the past two months, ETS has been operational and undergoing in use testing. As the students in CS 201 had already learned to use batch, it was decided to make ETS available on a voluntary basis only. Since there is a severe shortage of DECTapes, it also seemed unwise to encourage many students to switch to ETS.

The effort has been primarily on completing the documentation and DECTape filing system. The latter is complete and currently being used. The documentation consists of four parts:

- (1) The internal documentation of the code.
- (2) The implementation manual which describes the algorithms and organization of ETS itself.
- (3) The system programmer's guide which describes the conventions and facilities used to write code which will be run by ETS without interpretation.
- (4) The student user's guide which describes the facilities available to students who are learning to program.

A fifth manual is being gathered which will be a companion to the system programmer's guide and will provide descriptions of utility programs written for use by system's programmers only. This manual will consist of the documentation submitted by the authors in support of their programs.

At the present time, items (1) and (2) are complete and (3) and (4) are about 85% complete.

Preparations have been made for the addition of 12K of core to the system. This will allow us to increase both the size of the monitor and the available user area. The monitor will be increased

from its current size of 6K to 8K primarily by adding additional buffer space. The user area will be increased from its present 5K to 12-14K with the extra space being available for resident debugging and instrumentation packages. Changes to the monitor are complete and we await only the arrival of the core.

The instrumentation and tuning of the monitor are just getting underway. We initially plan two basic types of measurement for the instrumentation:

- (1) We will place counters in individual code segments and use the repetition counts to determine performance.
- (2) We will set up standard job mixes and measure the turnaround or throughput under different loading situations.

In order to give a feeling for the basic operation of ETS from the view point of a student user, we include as appendices to this report portions of the user's manuals for some system routines.

12.6 Magnetic Tape System for ETS (MAGETS)

MAGETS is a multi-user DECtape filing system which provides the users of ETS with permanent storage for their source and object files. Its capabilities are limited to copying between disc and tape. The MAGETS filing system consists of four different programs:

- (1) Formatter - Sets up the directory on the tape including a tape identification consisting of the tape name and its read and write passwords.
- (2) Mounter - Each time a DECtape is mounted on the drive it is necessary to inform the system that the tape exists. This is done via the MAGETS mount program. The mount program copies the directory to disk and checks to see if

the correct tape is mounted. If it is not, the user is notified so that he may mount the correct tape.

(3) MAGETS - MAGETS is the heart of the MAGETS tape filing system. It does all the transferring between disc and tape and also provides directory functions to delete and rename files and to list the directory.

(4) Dismounter - The MAGETS dismount program must be run each time a DECTape is to be removed from the drive. Its function is to write the directory back onto tape.

Storage Format of MAGETS

MAGETS stores up to 59 files on one DECTape. Files are stored on non-contiguous blocks, thus avoiding any garbage collection problems. To avoid the problem of the tape drive having to back up because DECTapes require three blocks to get up to speed, even blocks are read forward and odd blocks are read in reverse. Also to help prevent this problem, an attempt is made to allocate blocks for a file with at least four blocks between them. A maximum of 573 256-word blocks are available for user files.

Protection

Because ETS users will depend on MAGETS to store the only copies of their source files, MAGETS must guarantee their protection. Since more than one student may store files on the same tape, read and write protection codes can be specified for each file and for the tape as a whole. Thus to use the tape a user must know either the read or write passwords or both. Protection codes are up to three characters long.

Besides the read and write protection codes MAGETS makes numerous checks to insure that someone has not removed a tape and replaced it with another or anything else that could cause data to be written onto the wrong tape.

Multi-User Capabilities of MAGETS

MAGETS was designed with the capabilities of multiple users accessing the same tape directories simultaneously. To do this, all directory functions are non-resident file processor routines. This provides the proper interlocks and lets multiple users gain access to the same tape. Since the DECTape controller can control only one DECTape unit at a time, requests for the use of the controller are queued. The user becomes inactive while waiting for the request to be honored. When it is his turn the tape is checked to see if the correct tape is mounted. If it is the transfer is initiated. The tape driver is written so that while information is being processed, the tape driver is searching for the next block. After a complete file has been transferred the next user is started immediately.

(Don Oxley, Jim Hart)

13. GENERAL DEPARTMENT INFORMATION

13.1 Personnel

The number of people associated with the Department in various capacities is given in the following table:

	<u>Full-time</u>	<u>Part-time</u>	<u>FTE</u>
Faculty	22	4	23.97
Visiting Faculty	3	0	3.00
Research Associates and Instructors	2	2	2.743
Graduate Research Assistants	0	66	33.08
Graduate Teaching Assistants	0	29	14.625
Professional Personnel	10	1	10.50
Administrative and Clerical	19	0	19.00
Nonacademic Personnel (Monthly)	18	1	18.50
Nonacademic Personnel (Hourly)	<u>0</u>	<u>59</u>	<u>14.76</u>
TOTAL.....	74	162	140.178

The Department Advisory Committee consists of Professor J. N. Snyder, Head of Department, Professors E. K. Bowdon, D. F. Cudia, K. W. Dickman, M. F. Faiman, H. G. Friedman, C. W. Gear, D. B. Gillies, W. J. Kubitz, D. J. Kuck, B. H. McCormick, R. G. Montanelli, S. Muroga, T. A. Murrell, J. Nievergelt, J. R. Phillips, W. J. Poppelbaum, S. R. Ray, E. M. Reingold, J. E. Robertson, P. E. Saylor, D. L. Slotnick, J. E. Vander Mey, D. S. Watanabe, and T. Wilcox.

During the first quarter, the following publications were issued by the laboratory:

Report Numbers

- | | | |
|-----|-----|---|
| No. | 24 | Noh, Killion, "A Numerical Solution of the Matrix Riccati Equations," January 20, 1972. |
| No. | 25 | Machado, N. C., "An Array Processor with a Large Number of Processing Elements," January 1, 1972. |
| No. | 26 | Miura, K., "An Introduction to the Analysis of Time Series," February 22, 1972. |
| No. | 28 | LaVine, G., "Applications of Computer Technology to the Working Drawings Phase of Architecture," March 1, 1972. |
| No. | 29 | Garber, J. A., "Some ILLIAC IV Algorithms for Signal Processing," March 1, 1972. |
| No. | 505 | Bowdon, Edward K., Sr., "Network Computer Analysis," 1972. |
| No. | 506 | Whaley, A. D., "A Failure-Tolerant System," February, 1972. |
| No. | 509 | Bowdon, Edward K., Sr., and W. J. Barr, "Cost Effective Priority Assignment in Network Computers," 1972. |

Theses

- | | | |
|-----|-----|--|
| No. | 491 | Pirnat, Charles R., "Observer Position Detector for the Stereomatrix 3-D Display System," February, 1972, (Ph.D.). |
| No. | 493 | Han, Joseph Ching-Chi, "Tree Height Reduction for Parallel Processing of Blocks of FORTRAN Assignment Statements," February, 1972, (M.S.). |
| No. | 494 | Lefler, Robert M., "On the Realization of Boolean Functions Using <u>And</u> and <u>Or</u> Elements," January, 1972, (M.S.). |
| No. | 495 | Baer, David Michael, "Subroutines in the Glypnir Compiler," January 15, 1972, (M.S.). |
| No. | 498 | Noh, Killion, "A Numerical Solution of the Matrix Riccati Equations," January 20, 1972, (M.S.). |
| No. | 499 | Machado, N. C., "An Array Processor with a Large Number of Processing Elements," January 1, 1972, (Ph.D.). |
| No. | 500 | Pumfrey, Marion Arthur, "QAS Question-Answering System," February, 1972, (M.S.). |

Theses Cont'd

- No. 501 Yasui, Toshio, "Conversion of Decision Tables into Decision Trees," February, 1972, (Ph.D.).
- No. 503 Swanson, Larry Alan, "Simulation of a Tree Processor," January, 1972, (M.S.).
- No. 504 Sanford, Wayne C., "An Interactive Syntax Analyzer," January, 1972, (M.S.).
- No. 507 Koch, J. A., "Item Analysis," March, 1972, (M.S.).
- No. 510 Bracha, Amnon, "Application of Continued Fractions for Fast Evaluation of Certain Functions on a Digital Computer," March, 1972, (M.S.).

13.3 Colloquia

"The ARPANET: an Experiment in Resource Sharing", by Professor Michael S. Sher, Center for Advanced Computation, University of Illinois, Urbana, Illinois, January 10, 1972.

"Reducibilities Among Combinatorial Problems", by Professor Richard M. Karp, Department of Computer Science, University of California, Berkeley, California, February 14, 1972.

"The Illinois System for Interactive Graphics, Modeling and Simulation", by Professor C. W. Gear, Department of Computer Science, University of Illinois, Urbana, Illinois, February 21, 1972.

"The Computer As Composer's Tool", by Professor James Divilbiss, Coordinated Science Laboratory, University of Illinois, Urbana, Ill., February 28, 1972.

"Universal Logic Modules", by Professor Franco P. Preparata, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, March, 1972.

"The State Concept in Parallel Processes", by Professor A. N. Habermann, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania, March, 1972.

"Bio-Information Mining: prospects for extracting information from cells organelles and macromolecules", by Professor Bruce H. McCormick, Department of Computer Science, University of Illinois, March, 1972.

13.4 Drafting

During the first quarter, a total of 192 drawings were processed by the general departmental drafting section:

Large Drawings	24
Medium Drawings	95
Small Drawings	27
Layouts	5
Report Drawings	16
Change Order Drawings	15
Misceilaneous	10
TOTAL.....	192

(M. Goebel)

13.5 Shop's Production

Job orders processed and completed during the first quarter of 1972 are as follows:

	<u>AEC 2118</u>	<u>AEC 1469</u>	<u>Other</u>
Machine Shop		7	3
Electronic Shop	1	32	6
Chemical Shop	1	20	1
Layout Shop	1	76	14

(F. P. Serio)

DEPARTMENT OF COMPUTER SCIENCE
GRADUATE COLLEGE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

REPORT REQUEST FORM

Report Number

Title

Fold and Staple as shown

NAME:

ADDRESS:

Fill out Mailing Label



NAME:

ADDRESS:

CUT ALONG THIS LINE

FOLD

Mail Room
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

Stamp

FOLD

LITHOGRAPHIC DATA ET		1. Report No. OUCDCS-QPR-72-1	2.	3. Recipient's Accession No.	
Title and Subtitle Quarterly Progress Report				5. Report Date	
				6.	
Author(s)				8. Performing Organization Rept. No.	
Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801				10. Project/Task/Work Unit No.	
				11. Contract/Grant No.	
Sponsoring Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801				13. Type of Report & Period Covered Quarterly Progress Report - Jan.-March '72	
				14.	
Supplementary Notes					
Abstracts An abstract is not applicable.					
Key Words and Document Analysis. 17a. Descriptors Not Applicable					
Identifiers/Open-Ended Terms Not Applicable					
COSATI Field/Group					
Availability Statement RELEASE UNLIMITED				19. Security Class (This Report) UNCLASSIFIED	
				21. No. of Pages 226	
				20. Security Class (This Page) UNCLASSIFIED	
				22. Price	

0.84
265

Topics

UIUCDCS-QPR-72-2

COO-1469-0211

COO-2118-0037

QUARTERLY TECHNICAL PROGRESS REPORT

APRIL, MAY, JUNE 1972

THE LIBRARY OF THE

NOV 2 1972

UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

Clasot 25-15



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

QUARTERLY TECHNICAL PROGRESS REPORT

April, May, June 1972

UIUCDCS-QPR-72-2

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

TABLE OF CONTENTS

	Page
1. CIRCUIT RESEARCH	1
1.1 BURP (Project No. 21)	2
1.2 APE (Project No. 25)	2
1.2.1 The APE Sensors	2
1.2.2 The APE Transmitter	4
1.2.3 The Power Supply Voltage Regulator for the APEs	7
1.3 PENTECOST (Project No. 31)	7
1.4 ERGODIC (Project No. 39)	10
1.4.1 Summary	10
1.4.2 Ergodic Generator	10
1.4.3 Processor	10
1.4.4 Arithmetic Unit	10
1.5 TELEMAZE (Project No. 41)	14
1.5.1 Progress Summary	14
1.5.2 X, Y Coordinate Detector	14
1.5.3 Coordinate Processing Logic	17
2. HARDWARE SYSTEMS RESEARCH	19
2.1 LASCOT (Project No. 9)	20
2.1.1 Operation of the Three-Color Laser	20
2.1.2 Deflection and Modulation System	20
2.1.3 Scanner	20
2.2 OLFT (Project No. 12)	20
2.2.1 Write-Gun Isolation Amplifier Modifications	20
2.3 SEMANTRIX (Project No. 24)	22
2.4 LINDA (Project No. 28)	24
2.4.1 Summary	24
2.4.2 Project Status	24
2.5 RASER (Project No. 29)	25
2.5.1 Summary	25
2.6 STEREOMATRIX (Project No. 30)	25
2.6.1 Perspective Transformer	25
2.6.2 Coefficient Generator	25
2.6.3 Cursor and Inverse Transformer..	27
2.6.4 Stereomatrix Display	28
2.7 SCANTRIX (Project No. 35)	28
2.7.1 Output Driver Circuit	28
2.7.2 Address Generation	29
2.8 FROG (Project No. 36)	29
2.8.1 The Simulator	29
2.8.2 Summary of the Run	29
2.9 BEACON (Project No. 38)	33

	Page
2.10	OPITRACK (Project No. 40) 34
2.10.1	General Description 34
2.10.2	Construction. 37
2.10.3	Principle of Operation. 40
2.10.4	Some Design Equation. 42
2.10.5	Development 44
2.11	OCOMO (Project No. 42). 44
2.11.1	Introduction. 44
2.11.2	Present Work. 48
3.	SOFTWARE SYSTEMS RESEARCH 52
3.1	Numerical Processes 52
3.1.1	DIFSUB. 52
3.1.2	Sparse Eigen Problem. 54
3.1.3	Auto-Restart. 56
3.1.4	Plot Package. 57
3.1.5	Item Analysis 61
3.1.6	FINDEX. 63
3.2	Illinois Graphics Computing System (IGCS).. 72
3.2.1	Overlay Supervisor. 72
3.2.2	A Drafting Language (ADL) 72
3.2.3	Gould Printer/Plotter Interface.. 73
3.2.4	ILLISM-E. 74
3.2.5	Numerical Package 75
3.3	Graphical Remote Access Support System. . 77
3.3.1	OS/8 Operating System 77
3.3.2	Information Retrieval 77
3.3.3	Monitors. 77
3.4	Computer Maintenance and Construction . . 79
3.4.1	Graphics-8 Hardware 79
3.4.2	Equipment Maintenance Log Summary . 80
4.	IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH:
ILLIAC III.	83
4.1	Principal Developments in Brief 83
4.2	Interactive Picture Processing. 86
4.2.1	General Orientation 86
4.2.2	Picture Processing Languages and Their Implementation. 88
4.2.2.1	Show-and-Tell 90
4.2.2.2	PAX Language. 94
4.2.2.3	SOL: Structure Operation Language. 95
4.2.3	SCAN/DISPLAY. 110
4.2.3.1	Description of Operable Equipment 110
4.2.3.2	PDP8/e Controller and Interface to the IBM 360/75. 112

	Page
4.2.3.3	Automated Microscope 112
4.2.3.4	Video Digitization. 112
4.2.3.5	46mm Scanner Conversion 113
4.3	Patter Recognition. 115
4.3.1	Covering Theory 117
4.3.2	Variable-valued Logic 119
4.4	Image Processing Theory 121
4.4.1	Resume of Image Processing
	Methodology 124
4.4.2	Scene Segmentation. 126
4.4.3	Automated Analysis of Texture . 128
4.4.4	Atlas-Defined Information Image
	Deformation 141
4.4.5	Shape Identification. 147
4.4.6	Structure Transformations . . . 149
4.4.7	Experiments in Image Parsing. . 150
4.5	Applications. 151
4.5.1	Image Processing in Automated
	Cytology. 151
4.5.2	Brain Mapping 157
4.5.3	Cytospectrometer. 158
4.6	Computer Systems Support. 162
4.6.1	IBAL Assembler. 162
4.6.2	Pattern Articulation Unit
	Development 165
4.6.3	Arithmetic Unit Development . . 167
4.6.4	Taxicrinic Processor Development 167
4.6.5	Processor Intercommunication. . 168
4.6.6	Scanner/Monitor Usage 168
4.7	Bibliography. 169
4.7.1	External Documents Issued . . . 169
4.7.2	Logic Drawings Issued 172
4.7.3	Engineering Drafting Report . . 172
4.7.4	Administration. 173
5.	THEORY OF DIGITAL COMPUTER ARITHMETIC 174
5.1	Study of Logical Organization for LSI
	Implementable Arithmetic Units. 174
5.2	Implementation of Continued Product
	Algorithms. 175
5.3	Automatic Evaluation of Some Elementary
	Functions Using Microprogramming. 177
5.4	Continued Fraction Arithmetic 178
6.	SWITCHING THEORY AND LOGIC DESIGN 181
7.	OFFICE OF THE SOUPAC CONSULTANTS. 186
8.	MACHINE AND SOFTWARE ORGANIZATION STUDIES 188

	Page
8.1 FORTRAN Parallelism Detection	188
8.1.1 The Analyzer.	188
8.1.2 Parallelism Exploitation and Scheduling.	193
9. COMPUTER SYSTEMS ANALYSIS	198
10. NUMERICAL ANALYSIS.	200
11. COMPUTER LANGUAGES FOR MATHEMATICS AND NUMERICAL ANALYSIS.	201
12. COMPUTATIONAL ASPECTS OF COMBINATORIAL ALGORITHMS..	203
13. NUMERICAL METHODS, COMPUTER ARITHMETIC AND ARTIFICIAL LANGUAGES	205
13.1 GIZMO	205
13.2 Design Criteria for a High Language for System Programming.	205
13.3 MIPS.	208
14. GENERAL DEPARTMENT INFORMATION.	211
14.1 Personnel	211
14.2 Bibliography.	212
14.3 Colloquia	214
14.4 Drafting.	215
14.5 Shop's Production	215

1. CIRCUIT RESEARCH

(Supported in part by the Office of Naval Research under Contract
N000 14-67-A-0305-0007, W. J. Poppelbaum, Principal Investigator.)

Summary

Bernard Tse reports that the BURP project is almost completed. Hubert Wo reports on the design of three APE circuits: The APE Sensor, the APE Transmitter and Modulator and the APE Voltage Regulator. Godvarish Panigrahi describes the photo-detector synchronizing circuit for PENTECOST. Jim Cutler reports the design and construction of ERGODIC is well under way with only the arithmetic unit remaining. Ed Pott explains the operation of the TELEMAZE Input Coordinate Detection and Processing Logic.

1.1 BURP (Project No. 21)

Since the last quarter, most of the parts ordered for the project have been received and construction of the repeater/restorer boxes is underway.

It was decided during the last quarter that the repeater/restorer stages should be housed in separate chassis boxes, each with its own power supplies. The circuit boards for the repeater/restorers have been built and tested. Six power supply boards, each capable of supplying ± 5 volts at 1 ampere were also built.

The remaining work that has to be done involves building the chassis boxes for the 6 repeater/restorer stations. Each station has two 64-pin patch-boards, one for the input bundle and the other for the output bundle. These stations are connected to one another by the 64 patch-cords that make up the bundle.

Bump should be completed during the current quarter.

Bernard Tse

1.2 APE (Autonomous Processing Element, Project No. 25)

1.2.1 The Ape Sensors

As mentioned in the previous report, the function of the APE sensors is to perform input data acquisition for the APE machine by monitoring the input variables and encoding the results into pulse width modulated signals acceptable to the APEs. Two of these sensors have actually been implemented. The physical parameter which they sense is the intensity of a light shown upon their "eyes". A block diagram of the light sensor for the APE machine is shown in Figure 1. Because the output of the light sensor must be sent to the APEs synchronously, a clock receiver is required as indicated in the figure. The synchronization signals manifest themselves as an extraordinarily wide pulse in the clock pulse

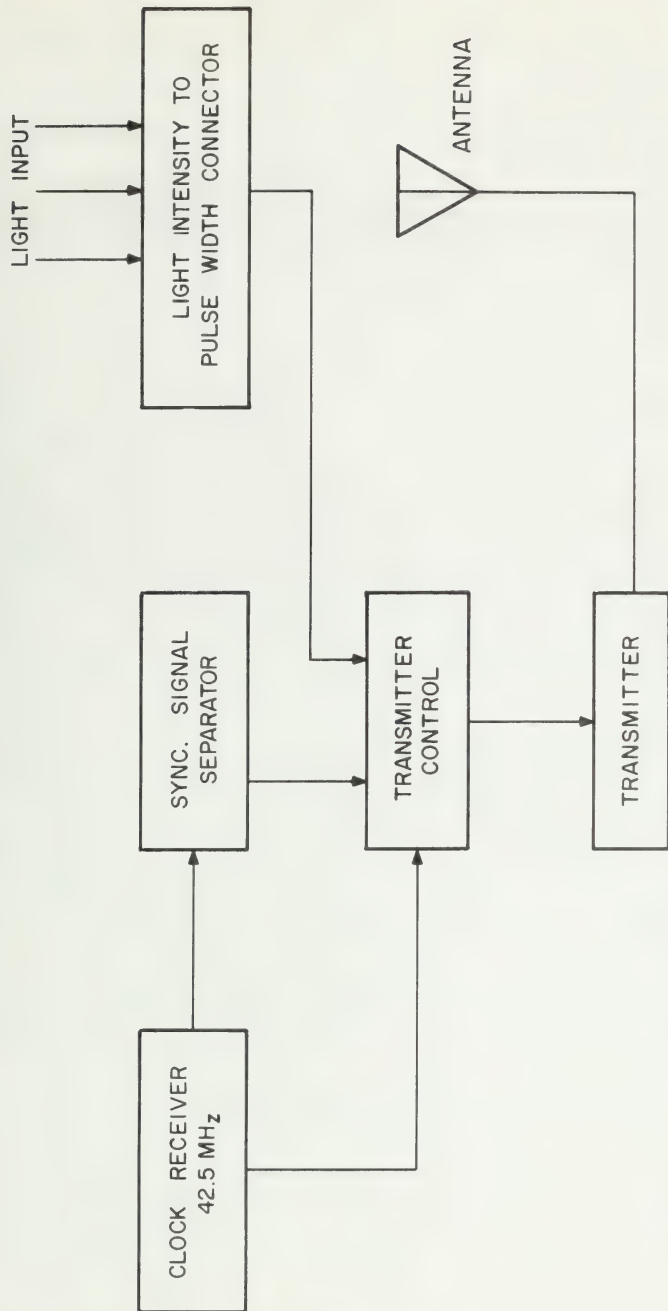


Figure 1. Block Diagram of Light Sensor

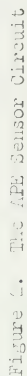
train. They are identified by the synchronization signal separating circuit. The light-intensity-to-pulse-width converter performs the actual sensing job. The transmitter control is designed to turn on the transmitter approximately 2ms prior to the beginning of transmission as required by the operation of the transmitter. It also turns off the transmitter after an RF pulse of appropriate width is transmitted.

Figure 2 shows the circuit diagram of the synchronization signal separator, the light-intensity-to-pulse-width converter, and the transmitter control logic. The synchronization signal separator utilizes a one-shot circuit to produce pulses with a standard width of $3.8\mu\text{s}$. These standard pulses are compared with the clock pulses by means of a D type flip-flop. A clock pulse having a width wider than that of the standard pulse, as the synchronization signal does, causes this D flip-flop to register a "1" at its output. Otherwise the output is a "0". The Q output of this D flip-flop then represents the synchronization signal and is used to trigger another one-shot whose duration is determined by the light intensity sensed by the photocell. The rest of the circuit shown in Figure 2 is for transmitter control. It is essentially a timer made up of counters and gates. It turns on the transmitter approximately 2ms prior to the next synchronization pulse coming in as required by the transmitter.

The clock receiver is identical to the one used in the APEs. It was described previously, and its description will not be repeated here.

1.2.2 The APE Transmitter

The transmitter is shown in Figure 3. It is a crystal-controlled low power switching transmitter. It was developed in this laboratory to satisfy the particular requirements of very low power consumption. The total power consumption at 15% duty cycle is about 6mW. A 2N2475 transistor is used to reduce



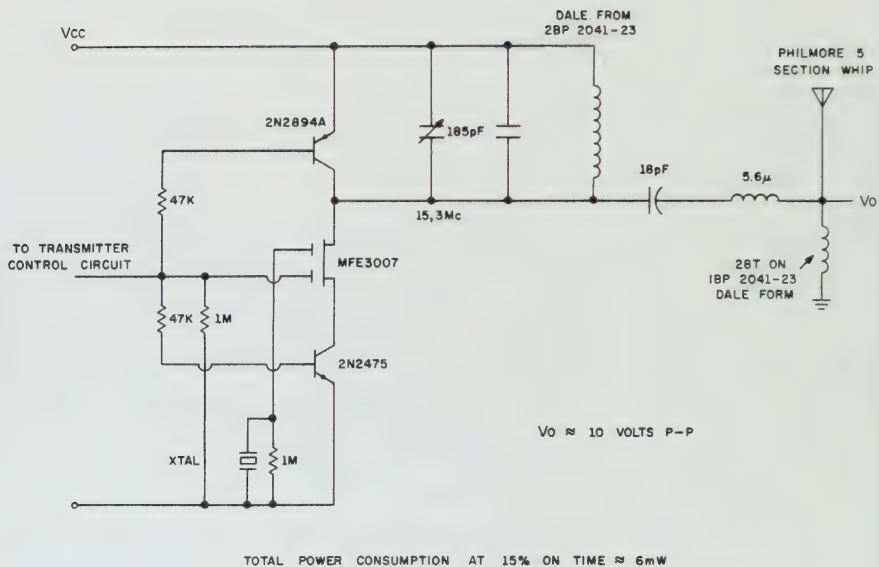


Figure 3. The APE Transmitter and Modulator

the stand-by power consumption while the 2N2894A is used to reduce the Q value of the tank circuit when the transmitter is shut off in order to produce an RF pulse with sharp decay.

1.2.3 The Power Supply Voltage Regulator for the APEs

As mentioned before, the power is remotely sent to the APEs. Therefore, power received by a particular APE may vary a great deal according to the physical location of the APE and its relative position with other objects. The function of the regulator is to guarantee a constant supply voltage to power the APE regardless of the fluctuation of the received power and independent of the variation of the load current for different modes of APE operation. The regulator circuit is shown in Figure 4. It makes use of a field effect diode which serves as a current source with a dynamic impedance of at least 800k Ω . The current source is used to regulate the current through the zener diode and thus reduce the variation of the reference voltage due to the variation of the input voltage to the regulator. In addition, a high beta transistor is employed in the output stage of the regulator to reduce the effect of variations of the load current on the reference voltage

Hubert Wo

1.3 PENTECOST (PENetration Tube Color System, Project No. 31)

A photo-detector circuit that detects the position of the filter wheel has been designed. It is shown in Figure 1. The light from an LED emitting in red passes thru the rotating filterwheel of red and green segments and is detected by a phototransistor. This signal is amplified and passed thru a Schmitt trigger in order to produce clean pulses. The output from the Schmitt trigger will be used to synchronize the camera and the subsequent switching circuits in the monitor.

G. Panigrahi

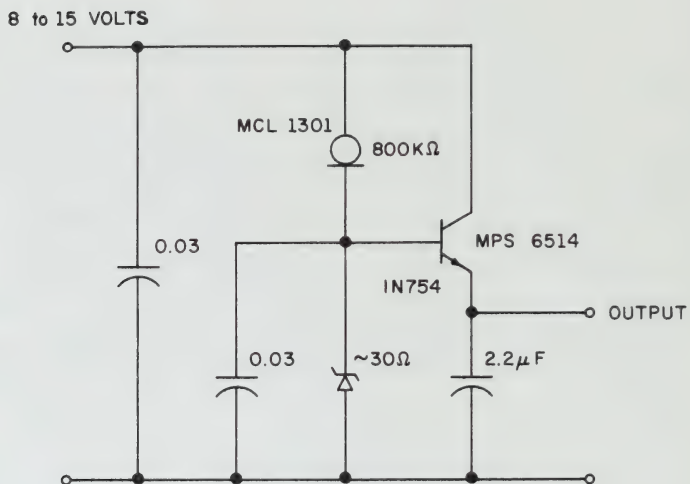


Figure 4. The APE Voltage Regulator

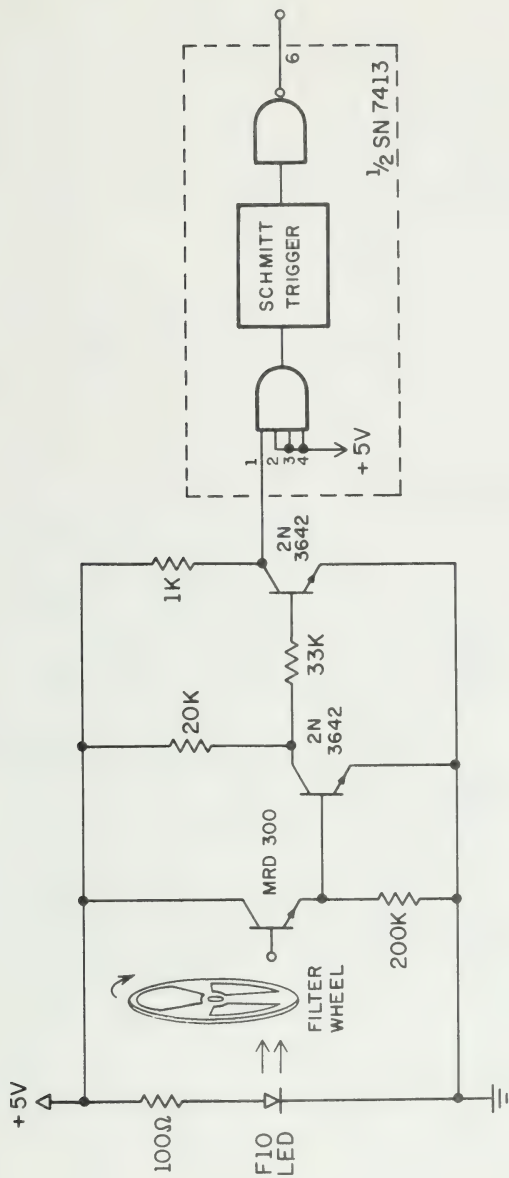


Figure 1. Filter Position Detector

1.4 ERGODIC (Project No. 39)

1.4.1 Summary

This project deals with a process that uses both bundle processing and stochastic processing in transmitting information and performing arithmetic operations. Further descriptive discussions were presented in previous reports.

1.4.2 Ergodic Generator

The Ergodic Generator is a 64-bit circular shift register. This part of the project has been completed along with a means for putting data (a digital fraction between -1 and +1) into the generator and a means for getting data out of this generator via a 64-wire cable.

1.4.3 Processor

The processor is capable of decoding the Ergodic Bundle into digital form. In addition, it can determine whether malfunctions have occurred on any of the wires. At this time all of the cards have been completed except the Nixie display, Nixie control and the lamp driver cards. All of the above cards have been designed and are presently being fabricated. This part of the project should be completed during the next quarter. The diagrams of the above cards are shown in Figure 1 through Figure 3.

1.4.4 Arithmetic Unit

The Arithmetic Unit performs arithmetic operations (multiplication, division, addition, subtraction, perhaps more) on two Ergodic bundles. No work has been done in this area since the processor is needed in order to do these studies. When the processor is completed, concepts for the Arithmetic Unit can be formulated and evaluated.

Jim Cutler

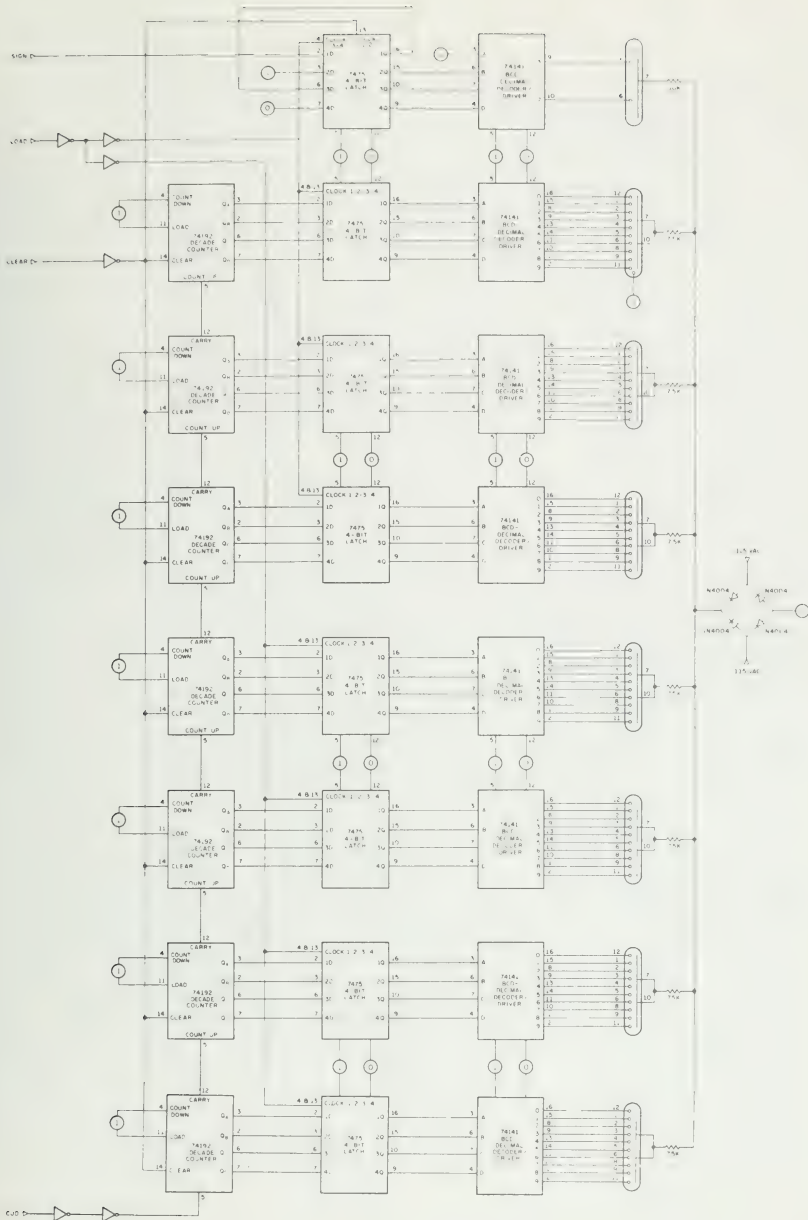


Figure 1. Nixie Display Circuit

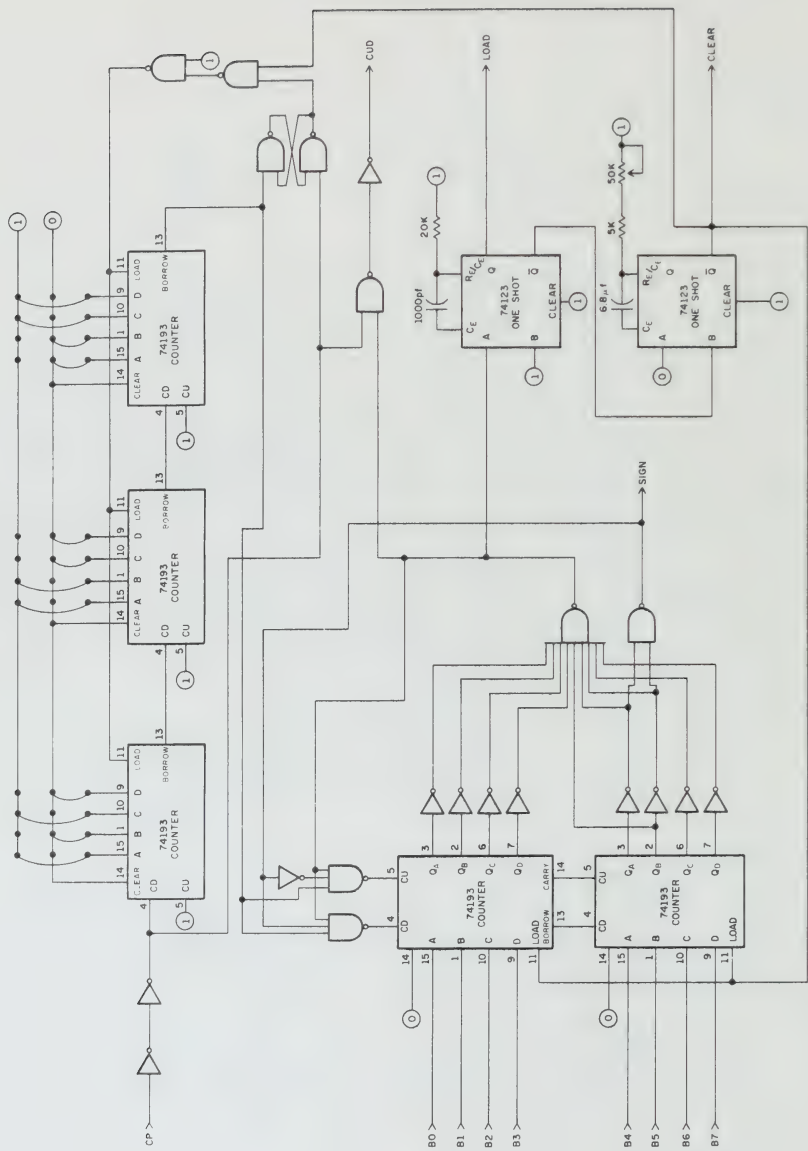


Figure 2. Nixie Control Circuit

1.5 TELEMAZE (Project No. 41)

1.5.1 Progress Summary

The primary work completed during this period was on the Input Coordinate Detection and Processing Logic. This logic serves as the main input component to the TELEMAZE System. This circuitry has been successfully constructed and mounted to the front of the local station television monitor as shown in Figure 1.

1.5.2 X, Y Coordinate Detector

The function of this logic is to detect the presence of the finger when placed upon the face of the CRT. The finger is used as a pen to generate the coordinates of the desired square used in creating a path for the vehicle to follow.

The hardware used to realize this function is shown in Figure 2. The heart of the system is a four-bit binary counter which provides sixteen addressable channels. That is, each axis can distinguish only one of sixteen lines. Each channel consists of one infrared light-emitting diode (LED) and one phototransistor. To enhance the system performance, lenses are also used. One lens is used for each LED and one for each phototransistor. These lenses are used to focus the infrared light beams. Each channel is enabled in sequence until one light beam is found to be obstructed.

The transmitting board determines which LED is to be switched on. A four-bit to sixteen-line decoder is used for this. Each LED is switched on by its corresponding 2N3638 transistor. The 2N2219 transistor is used as a constant current source to control LED current.

In the receiver section, incident light on each phototransistor is used for beam detection. The multiplexer gates only one phototransistor output on at a time. This serves to prevent emissions at nearby LEDs from false triggering adjacent channels. If the phototransistor is conducting

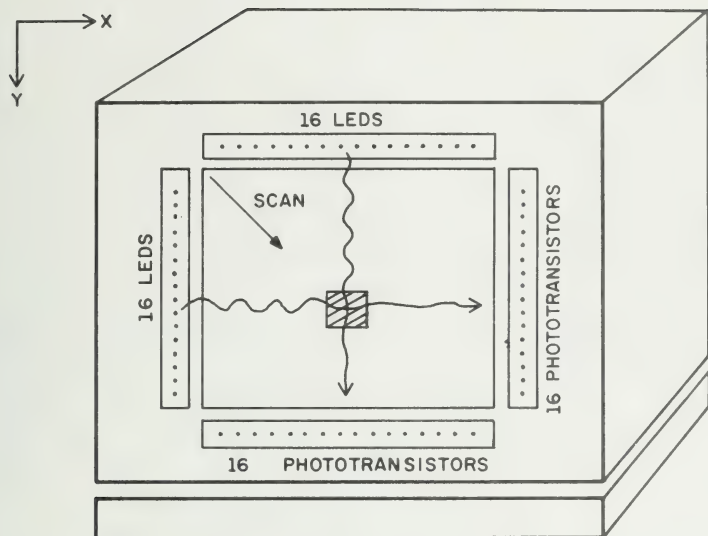


Figure 1. TELEMAZE Local Station

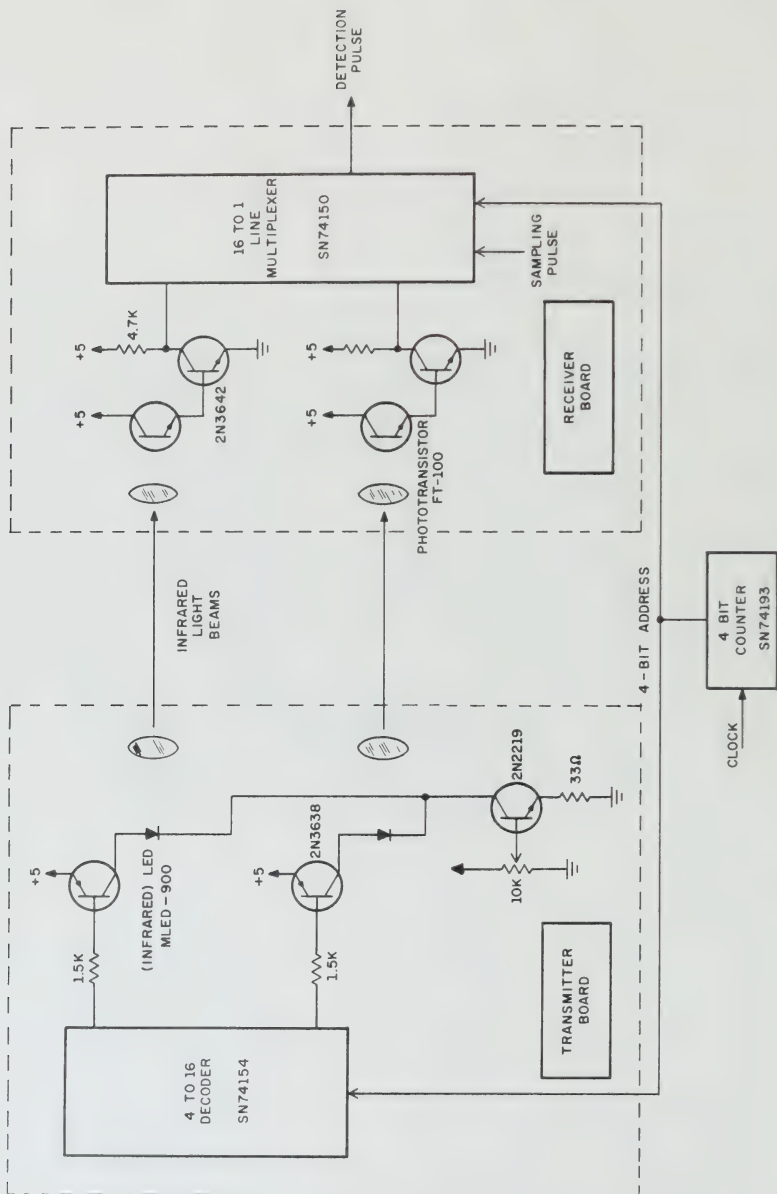


Figure 2. TELEMAZE Coordinate Detection Logic

a zero output is obtained. This technique outputs a pulse from the multiplexer only when a light beam has been broken.

The major limitation to speed in this system is the phototransistor risetime. To overcome this and reduce the cycle time, a sampling pulse is used. The function of this pulse is to allow the phototransistor to reach the conducting state before the channel is tested.

1.5.3 Coordinate Processing Logic

Once either coordinate has been detected, a pulse is sent to the coordinate processing logic. The current address of the 4-bit counter is then retained in its respective register as shown in Figure 3. In addition, a flip-flop is set to denote one coordinate has been found. The other coordinate axis circuitry will continue to scan until the other detected pulse is obtained. It is the coincidence of both the X and Y axis flip-flops that will enable the 8-bit output register to be gated on (at the end of the cycle). The output from this register will be the X, Y address in binary form. It will remain in this register for further processing for at least one cycle time.

Edward Pott

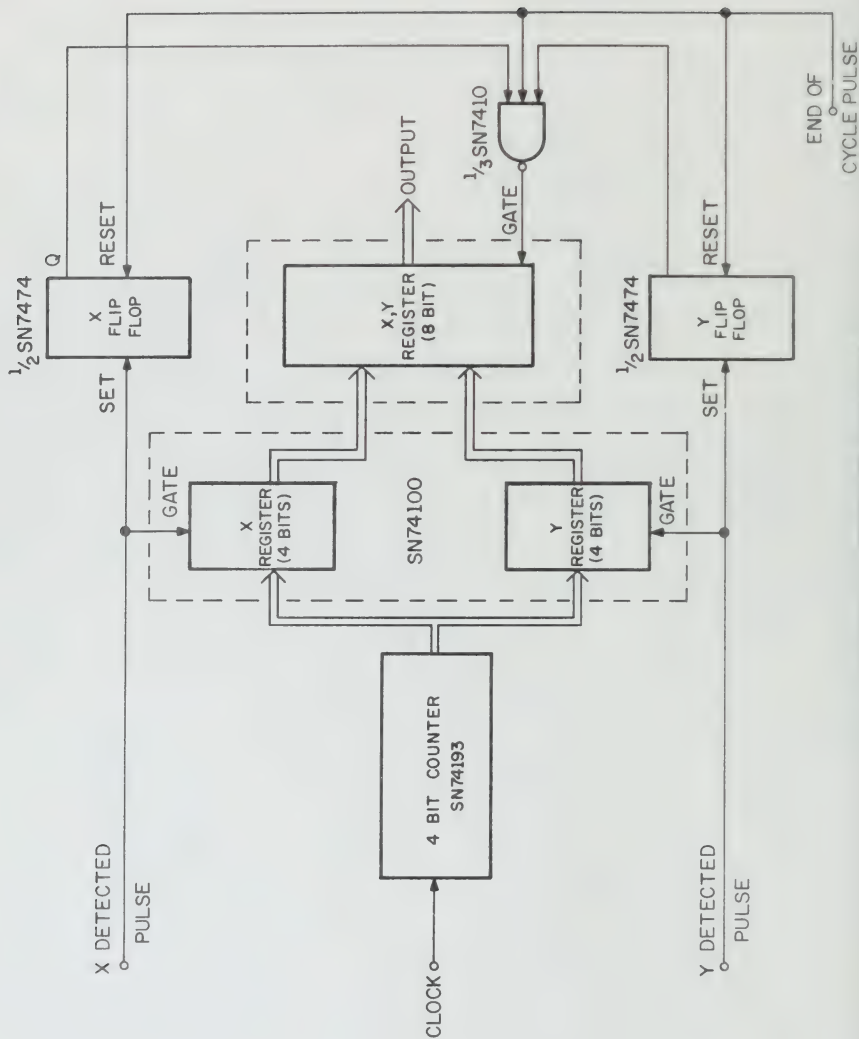


Figure 1. TELETYPE Coordinate

2. HARDWARE SYSTEMS RESEARCH

(Supported in part by the Atomic Energy Commission under Contract
US AEC AT(11-1) 1469, W. J. Poppelbaum, Principal Investigator.)

Summary

Mehernosh Cooper reports on the operation of the three-color laser and the scanner for the LASCOT project. Doug Sand describes the modifications which have been made to the OLFT write-gun amplifier. Dick Blandford extolls the success of the LINDA model. Tak Katoh reports the completion of RASER. Steve Whiteside and Shiv Verma report that the Stereomatrix Coefficient Generator is operational but for a few problems, the Transformer and the Cursor are complete, the Inverse Transformer is in the final stages and the Display can be completed once the laser power supply modifications are completed. Sik Yuen explains the operation of the output driver circuit and the address generator for SCANTRIX. Dev Bose reports successful operation of the simulator for FROG. Martin Jer describes the storage and driver circuit for the BEACON system. Mohamed El-Sonni describes the latest version of OPTITRACK. Last, Robert Budzinski explains a new project, OCOMO (Optical Correlation Modulation).

2.1 LASCOT (Large Screen Color Television, Project No. 9)

2.1.1 Operation of the Three-Color Laser

The Spectra Physics Model 164, mixed gas (Ar, Kr) laser has been installed. The various colored lines of interest (red, green, blue) have been obtained from the laser beam by means of a series of prisms.

2.1.2 Deflection and Modulation System

A raster was obtained using the LASCOT deflection system. The modulators were driven from the output of a three-stage binary counter in order to produce a color bar pattern consisting of eight different colors in the raster.

2.1.3 Scanner

The LASCOT scanner was used to detect the image on the screen of a color television set. It was observed that the decay time of the phosphor (to 10%) was of the order of 50 μ s. This necessitated that the vertical deflection mirror be synchronized with the television vertical deflection. After synchronization of the vertical mirror was established, the detected image was found to be of very poor quality because of the very short persistence of the tube phosphors.

Several possible solutions to this problem are being explored so that the quality of the picture can be improved.

Mehernosh Cooper

2.2 OLFT (On-Line Fourier Transform, Project No. 12)

2.2.1 Write-Gun Isolation Amplifier Modifications

The isolation amplifier which drives the cathode of the write-gun has been modified in several ways. First, the filament power was changed to an adjustable, regulated, direct current source. Second, the isolation coupler was changed from an rf carrier and transformer to a pair of photon-coupled isolators and a blanking clamp was added.

The filament power was previously supplied by a 7.5 VAC winding on the main power transformer. This supplied sufficient power for operation at normal cathode temperatures. However, the increased capacitance of the cooled crystal requires considerably more cathode current than that available at the nominal operating point. By increasing the cathode temperature, emission can be increased by a factor of 10 to 100, although cathode lifetime is reduced to about ten hours. To supply the needed filament power, an adjustable, regulated, direct-current source has been included. The current can be switched remotely from standby (400mA) to the desired operating value for short periods of time. Standby operation extends the useful lifetime of the cathode.

The video coupling method was changed because the old system (a 100 MHz amplitude-modulated carrier with transformer coupling) caused a variety of problems. First, the system was never truly "dc-clamped", but produced "sagging" dc levels when operated with the frame-gating required by the long storage time of the cooled crystal. Second, some rf radiation, although small, was "detected" by the pn-junction temperature sensors, causing erroneous temperature indications. The new coupling system uses one of the photon-coupled high-voltage isolators described in the previous report. Restoration of the dc level is done by the "unblank" circuit described below.

On several occasions transients, apparently due to AC-main power switching, have caused catastrophic overvoltage conditions characterized by uncontrolled write-gun current bursts which deposit enough charge on the crystal surface to produce a voltage several times the half-wave voltage of about 200 volts. This high voltage produces large piezoelectric forces, and on each occasion several cracks appeared in the crystal. Further, since this voltage

is too high to allow erase-gun operation, the only method of "erase" was to warm the crystal up to about -20°C so as to reduce the charge-decay time-constant to a reasonable value. Since these transient bursts are so undesirable, a write-gun clamp has been included in the isolation amplifier. This clamp is controlled by a digitally driven high-voltage photon-coupled isolator which transmits an "unblank" signal about $52\mu\text{s}$ long for each horizontal line being written. This unblank pulse gates a clamp on the G1-voltage so as to allow cathode current to flow and, in addition, provides for dc-level restoration during blank intervals. There is also a one-shot which limits the continuous unblank time to about 100mS (three frames) followed by a 1-minute lockout, which should limit catastrophic transients.

Douglas Sand

2.3 SEMANTRIX (Project No. 24)

The design of the system has been completed and most of the subsystems are under construction. With the completion of the design the instruction set for the machine has been finalized. The instructions are sent either from a standard ASR30 Teletype or from some suitable data processor which can send and receive information in Teletype format.

There are four basic instructions that the machine responds to:

- 1) The coordinates of any cube can be requested as follows;

Send: $N\ mn\ Rt$

Where mn is a two-digit octal number representing the code number of the particular cube whose coordinates are required.

A reply will be generated by Semantrix as follows;

Receive: $Lf\ x_2x_1x_0\ y_2y_1y_0\ Rt\ Lf$

Where $x_2x_1x_0\ y_2y_1y_0$ is a 6-digit octal number uniquely describing the location of the center of the cube in question on the quadrupled

grid of the machine. If the message sent has a syntax error the following message will be received: Lf E Rt Lf

Lf and Rt are nonprinting characters which cause a line feed and carriage return, respectively, when received by the Teletype.

- 2) The electro-mechanical Arm-Hand effector can be instructed to move to a point on the quadruled grid, as follows;

Send: C $y_2 y_1 y_0$ $x_2 x_1 x_0$ Rt

Upon execution of this instruction, Sematrix generates the following reply:

Receive: Lf Bell

Where the non-printing character bell actuates the bell on the Teletype.

A syntax error in the message sent to Sematrix causes the same reply as in 1).

- 3) The electro-mechanical hand on the end of the arm can be made to lower and raise as follows:

Send: LE Rt (lower)

RE Rt (raise)

- 4) It can be made to open and close as follows:

Send: EO Rt (open)

EL Rt (close)

For both instructions 3) and 4) a reply of Lf bell indicates Sematrix has successfully completed the instruction. A reply of Lf E Rt Lf indicates a syntax error in the message sent.

In addition to the input format as described above the following input to Sematrix is acceptable;

* N * 3 * 2 * Rt

Where * can be the null string or any string of Teletype characters, the only restriction being that it doesn't contain another allowed input sequence. Formally, if, $W = W_1 \dots W_n$ Rt is an allowed input. $A_1 W_1 A_2 W_2 \dots A_n W_n A_{n+1}$ is too, provided $A_1 A_2 \dots A_{n+1}$ doesn't contain an allowed input. If N3297 Rt was input the interpretation would be that the coordinates of block number 32 are requested. However, if N32EO Rt was input, it would be interpreted as a syntax error because N32 Rt and EO Rt are both allowed inputs. The don't care assignment of the stars gives an extended mnemonic facility and also allows for a flexible input format. For example instead of LE Rt, LOWER Rt can be sent, to lower the hand.

Trevor Mudge

2.4 LINDA (LINE Drawing Analyzer, Project No. 28)

2.4.1 Summary

The goal of project LINDA is to build a simple machine which can recognize a small vocabulary of line drawings. A given drawing is displayed on an oscilloscope via a flying spot scanner; the drawing is identified by knowing its simple parts and their relationship to each other. A more detailed description of the project is given in the previous quarter's report which includes a system block diagram.

2.4.2 Project Status

During the past quarter a simplified model of the system was completed. The model is presently being used to determine how much of a problem video noise will be and what can be done to null its effect. The model also allows thorough testing of the one-line video delay (see previous report) and the separation algorithm. The circuitry of the model allows LINDA to process line drawings having no more than three simple parts in a vertical row and no more than two video pulses on a single raster scan line. The model works

rather well. In addition, any one of the three parts of the line drawing may be isolated on the screen and shaded in or erased. Noise on the video line is a serious problem as was expected. Similar problems were encountered with the Tri-Color Cartograph and ORBIT. During the next quarter the noise problem will be given particular attention. Further, the complexity of the model for separation and shading of line drawings will be increased.

Dick Blandford

2.5 RASER (RANdom to SERial Converter, Project No. 29)

2.5.1 Summary

During this quarter the RASER project was completed and the final report issued (DCS report 515)

Tak Katoh

2.6 STEREOMATRIX (Project No. 30)

2.6.1 Perspective Transformer

The perspective transformer has been completed and the details of this unit have been published as DCS Report No. 532.

2.6.2 Coefficient Generator

This quarter has been devoted in debugging this unit. Oscillations in TTL Exclusive-Or circuits were caused by bad ground connections between racks. The twenty-foot control box cable may be a contributor to the problem, although additional grounding seems to have stopped the oscillations. There was some difficulty with the forward and backward buttons on the control box causing a reset when released. This has been largely eliminated by reducing the current that the buttons must switch from 5 ma. to 0.5 ma. The observer can rotate, translate and magnify the three-dimensional analog pattern obtained from PAGAN. Thus, all the features of this unit are working satisfactorily. However, incremental rotation in the machine is limited to five degrees due to

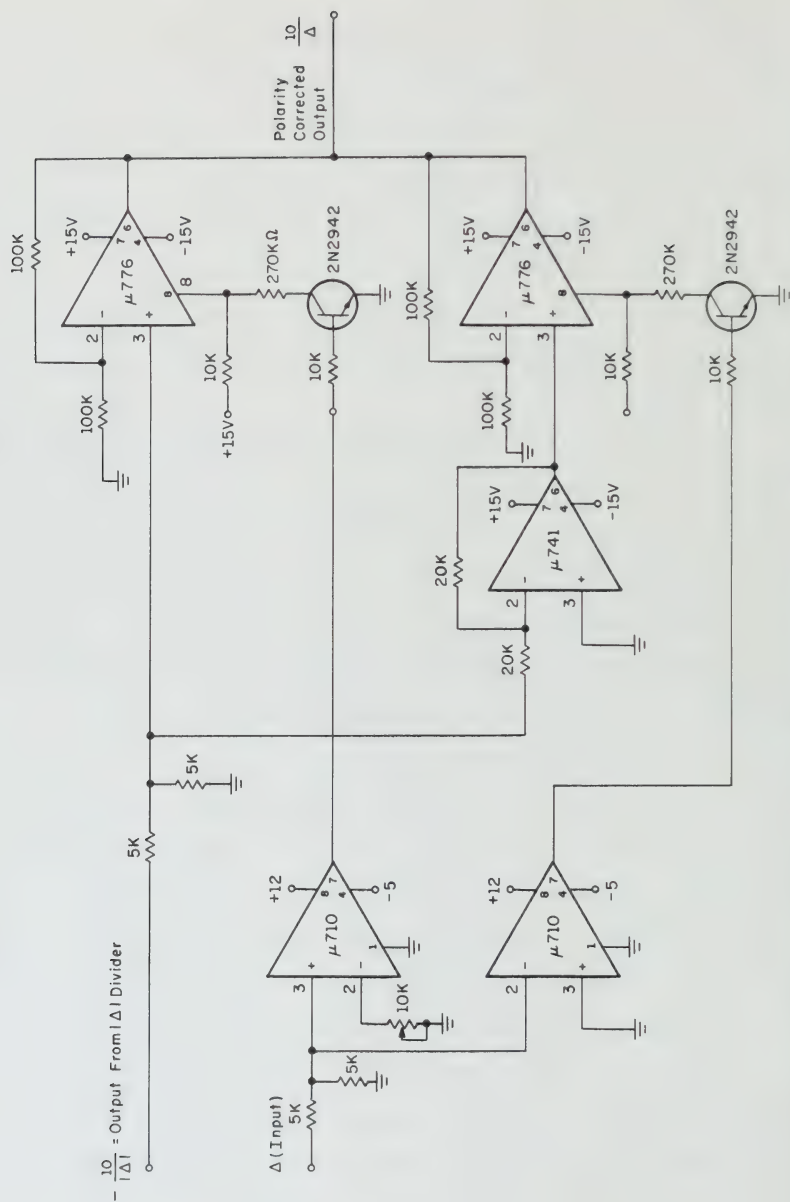


Figure 1. Δ Polarity Correcting Circuit

the 9-bit accuracy of the system. Efforts will be made to achieve 0.2 degree rotation per frame with modifications to the existing system.

2.6.3 Cursor and Inverse Transformer

The Cursor portion of the system is working. The observer can move the three-dimensional cross in the display volume by moving the joystick on the control box. The coincidence detection is also working. At this time, the cursor is also rotated and translated during rotation of the picture as the inverse transformer is not yet working.

The inverse transformer should be operational soon as all of the required circuits for this unit have been designed and tested. During inverse transformation, this unit must generate the determinant (Δ) of the transformation matrix. Then, a divider is used to generate $\frac{1}{\Delta}$. However, the stability of the divider limits Δ to being greater than one in magnitude and always negative in sign. Therefore, the inverse transformer requires a circuit which generates the right polarity of $\frac{1}{\Delta}$ after the divider gives $-\frac{1}{|\Delta|}$. The scheme of Figure 1 accomplishes this. A Fairchild $\mu A776$ Op. Amp. is used in the circuit. This Op. Amp. is turned ON/OFF by switching the 2N2942 ON/OFF. During the ON period the 2N2942 supplies the $50\mu A$ bias current required by the Op. Amp. for its linear operation. When the 2N2942 is off, the Op. Amp.'s current sources are off, and thus it is inoperative. Fairchild $\mu A710$ comparators are used to determine the polarity of Δ .

It is hoped that all the functional units of the STEREOMATRIX system will be working by the end of this quarter.

2.6.4 Stereomatrix Display

The strong RFI produced by the voltage control SCR's in the laser power supply is sufficient to stop all operations in the coefficient generator. The strong one MHz noise apparently supplies continual reset pulses to the coefficient registers. Some attempts at reducing the noise were only partially successful. Hence, another heatsink complete with cooling water tubing was constructed and partially installed. When ordered parts arrive the improvements will be completed and the supply will have double its previous dissipation capability (new capability will be 5KW). The voltage control SCR's will then be removed. Additionally, the old 100 volt transformer-in-series start circuit will be replaced with a 600 volt transformer and a single diode wired in parallel with the laser tube. This is to eliminate inductive spikes applied to the current regulator by the old starting method.

The optics have been mounted on a new flat plate table and will be rigidly bonded in place when the laser is working and alignment can be optimized.

Shiv Verma and Steve Whiteside

2.7 SCANTRIX (SCANNed maTRIX, Project No. 35)

2.7.1 Output Driver Circuit

An LED driving circuit (see Figure 1) has been designed which has a storage time longer than 33.3msec (30 pictures/sec for TV). The circuit consists of a FET (2N5939) whose drain current is controlled by the gate voltage. When the analog gate (CD4016AE) is actuated by a positive pulse, the gate is open allowing the video to pass through to the capacitor. This voltage is stored in the capacitor and is used to control the drain current of the FET, lighting the LED correspondingly. Storage is achieved by a small capacitor because of the very high input impedance ($> 10^9$ ohms) of the FET and the very high output

impedance of the gate in its off state. The diode (TI51) and the 5V supply at the source of the FET biases the FET so that the FET is off for gate voltages below zero.

The circuit in Figure 1 also shows a NOR gate whose X, Y inputs correspond to the horizontal and vertical positions of the LED in the matrix. The SN7407 is a buffer which interfaces the TTL circuits to the COS/MOS circuits.

2.7.2 Address Generation

To address the LED's in the matrix, two address demultiplexing units are required; one for the horizontal and the other for the vertical. Figures 2 and 3 show such a unit. Note that only one unit is shown, the other being identical. When the system is turned on, the RESET is actuated to set $X = 0$, $Y = 0$. As the system clock comes in, the SN74163 binary counter counts the pulses providing the 1-of-16 demultiplexers with the appropriate inputs. The "carry" output from the SN74163 is used to drive a ring counter made up of SN7474's for enabling the appropriate demultiplexers. The SN74123 is used to provide a small delay so that none of the outputs from the demultiplexers are omitted.

Sik Yuen

2.8 FROG (Project No. 36)

In this quarter a simulation of FROG was implemented and the results analyzed, a summary of which is given below.

2.8.1 The Simulator

The simulator is a PL/I program. It consists of a main program (procedure) FROG and a few subroutines to simulate the MIU, (Memory Integration Unit), the MIS (Memory Integration Subunit), etc. Function procedures simulate the logic elements S, E, P, T, etc. (see previous reports). The main procedure

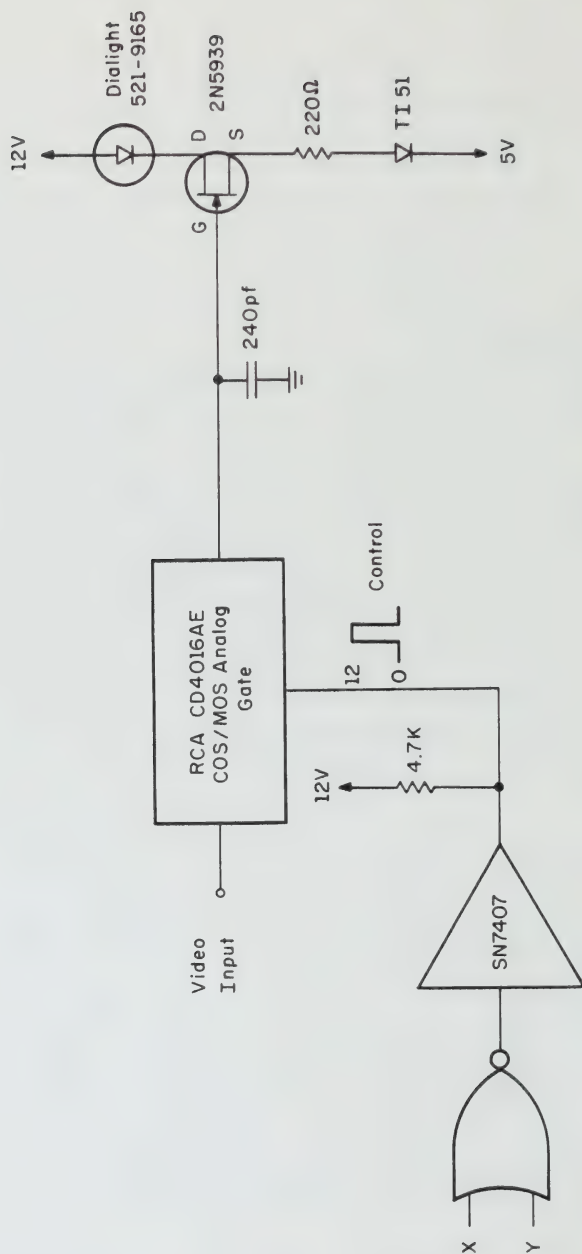


Figure 1. LED Driving Circuit

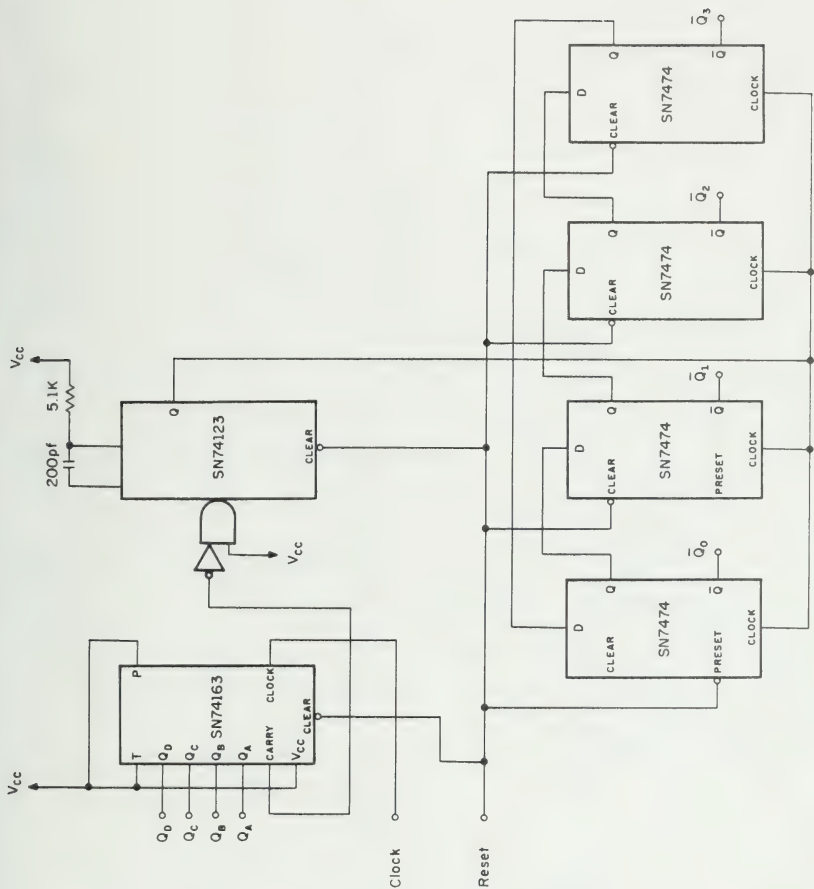


Figure 2. Demultiplexer Actuator and Addresser

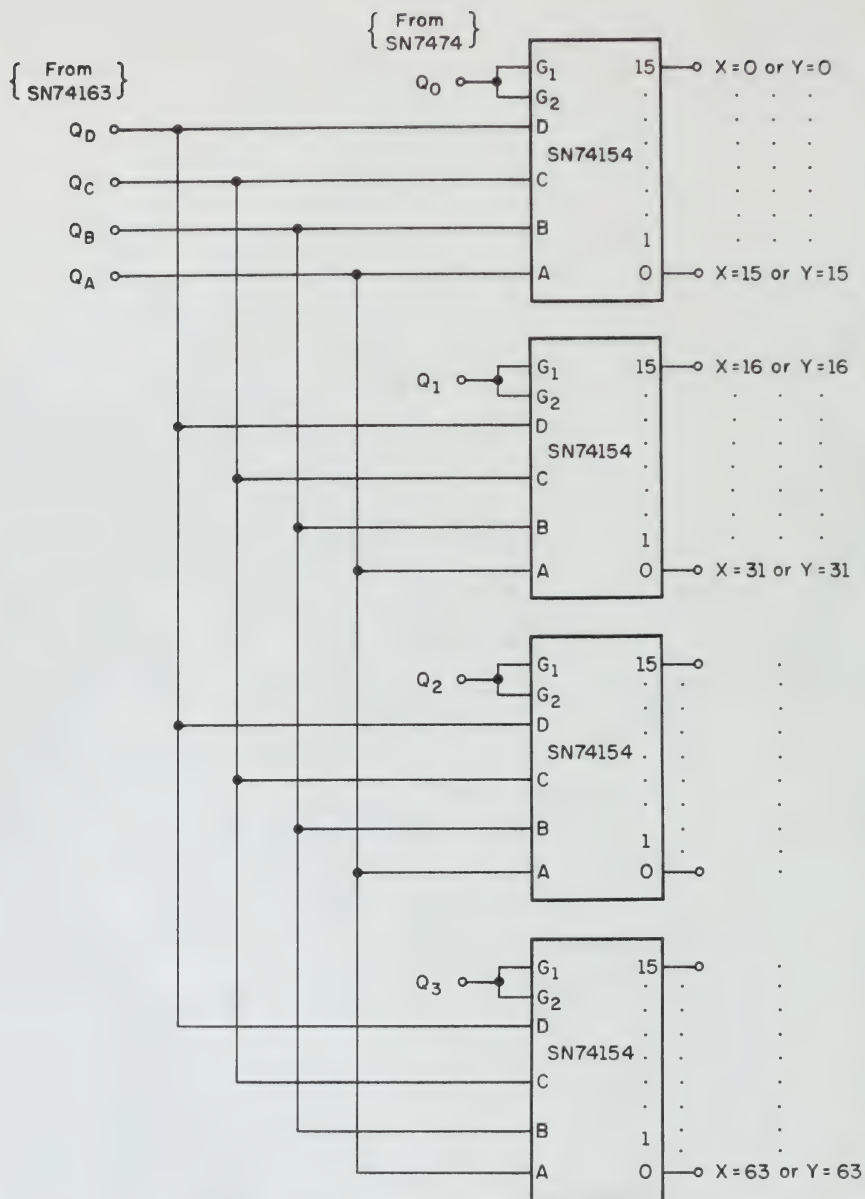


Figure 3. Row and Column Address Demultiplexer

FROG includes the program to generate the stimulus in the various regions in a random sequence. The frequency of appearance of a given stimulus in a given region can be specified a priori.

2.8.2 Summary of the Run

The run reported on here had 200 trials. In each trial a stimulus was presented to FROG and its reaction noted. The results show that FROG was able to correctly learn the various stimuli. However, as expected, there were errors in judgment during the initial trials. There were two distinct types of errors made by FROG. The first was deciding to feed on a bad tasting bug or on a predator. These errors were easily corrected, for the feeding resulted in additional learning which alleviated the problem. The second type of error occurred when a good-tasting bug was avoided. This error was difficult to correct; with feeding inhibited additional learning was not possible. Only starvation forced feeding helped reverse the situation for these errors.

If two or more stimuli had the same value for a particular visual characteristic in a given level and if this value was the largest in that level, the recognition was often erroneous. This was primarily because the approximations made in the various nonlinear operations made the memory structure and memory integration slightly different from the basic scheme laid down in the Quarterly Report July - September 1971. Necessary changes are being made to alleviate this difficulty.

Dev Bose

2.9 BEACON (BEAm CONTROL, Project No. 38)

For the past quarter work has been mainly devoted to developing the storage and driver cells which are used to amplify the signals from the photo-transistors. The amplified (and optionally stored) signals are used to drive

the display LEDs. The cell is equipped with a "frame freeze" or hold network that maintains on the display the information in a chosen frame for a period of 2 to 3 minutes. The circuit is shown in Figure 1.

Effort is now being directed toward developing a sequential network in order to store the picture line by line. Line by line storage is necessary in order to compensate for phosphor decay.

Martin Jer

2.10 OPTITRACK (OPTical TRACKer, Project No. 40)

2.10.1 General Description

OPTITRACK is an optoelectronic tablet system which encodes the position of a light pen on a glass tablet into its binary equivalent by optoelectronic means. (See Figure 1) It consists of the following parts:

- 1) A light pen as a point source of light.
- 2) A glass tablet as the writing surface.
- 3) A semireflector, which transmits a portion of the light incident and reflects the rest.
- 4) Two cylindrical lenses: one for x and the other for y encoding.
They are positioned such that the focused line images are displaced from the major axis by distances corresponding to their point source coordinates on the table.
- 5) Two encoding discs which are made of transparent material on which the position is encoded as clear and dark lines. They are subdivided into sectors. Each sector represents a single bit of the encoded word, which, in turn, represents the coordinate in digits. There are also two tracks of holes used for providing clocking and synchronizing signals. They are sensed by the sense head.

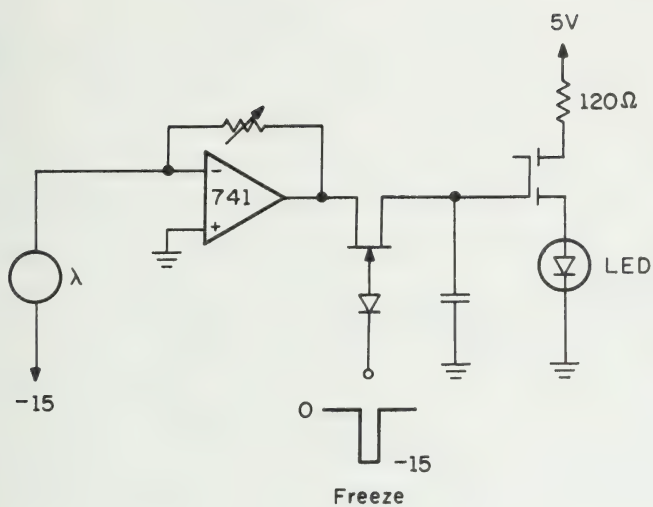


Figure 1. Driver Circuit

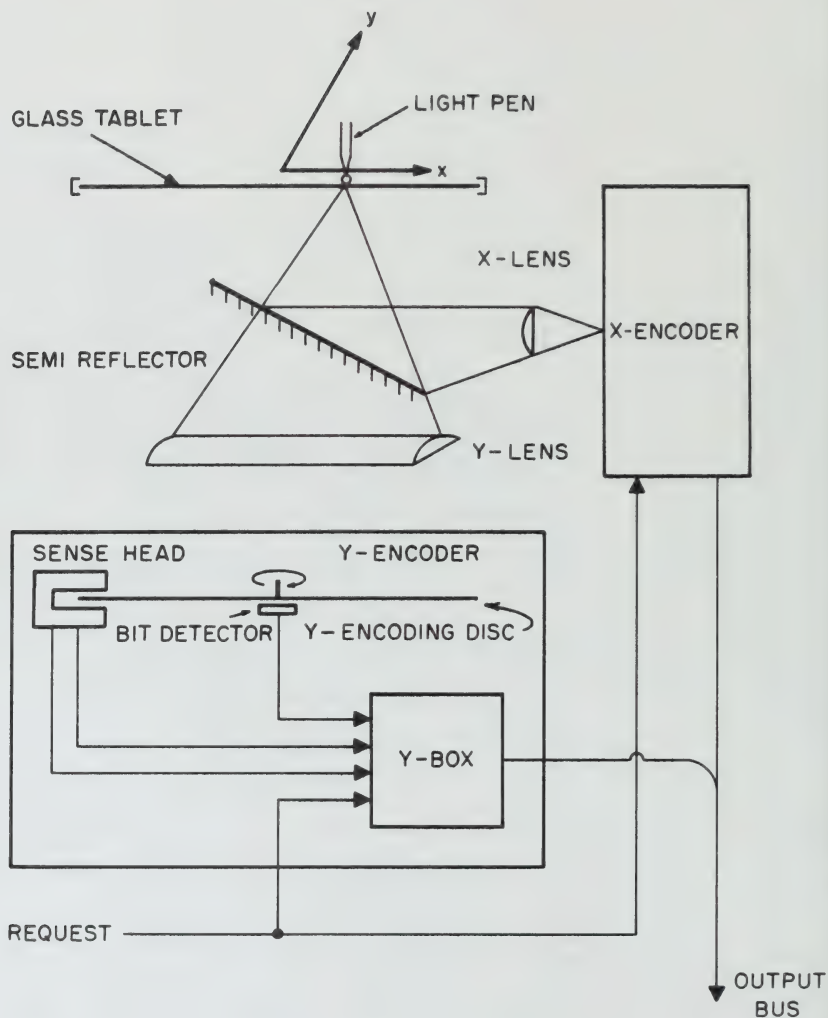


Figure 1. OPTITRACK Schematic Diagram

- 6) A line detector for each encoder is used to detect the existence of light transmitted through the corresponding disc.
- 7) Two boxes, x and y, consisting of registers and their associated logic.

2.10.2 Construction

Described below is the construction of the light pen, the encoding discs and the sense heads.

For optimum emitting and detecting efficiency an infrared emitting diode has been chosen as the light source. A pressure sensitive mechanism will be designed such that the diode will emit only while writing on the glass table surface. See Figure 2.

The discs (see Figure 3) are subdivided into N_s sectors such that N_s is an integral multiple of k , the word length required. A sector S_{m_1} represents a bit b_i where

$$m_1 = n_o k = i \quad 0 \leq i \leq k-1, \quad 0 \leq n_o \leq \frac{N_s}{k} - 1.$$

There are also two tracks of holes, one track for the clock and another for the synchronization, located at the perimeters of the discs. The total number of clock holes is equal to the total number of sectors while the number of synchronization holes is $\frac{N_s}{k}$. The discs are driven by a motor. By sensing the clock and the synchronization holes the operation of the tablet system will be asynchronous in nature once we set the correct initial state.

The sense head (Figure 4) consists of a light emitting diode placed on one side of the disc and two photodetectors on the opposite side to detect the clock and sync. signals. The clock will be used to strobe the detected signal from the line detector and to clock the register. The sync. signal will mark the beginning of each train of k bits (one word).

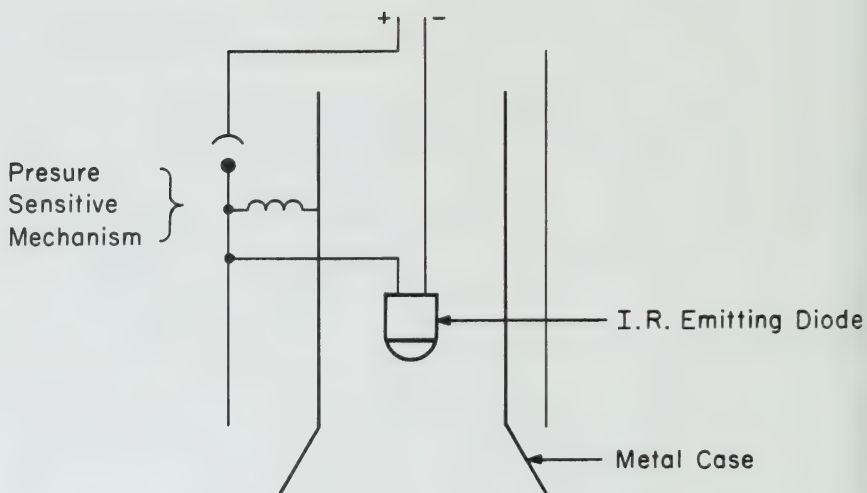


Figure 2. Pen Schematic Diagram

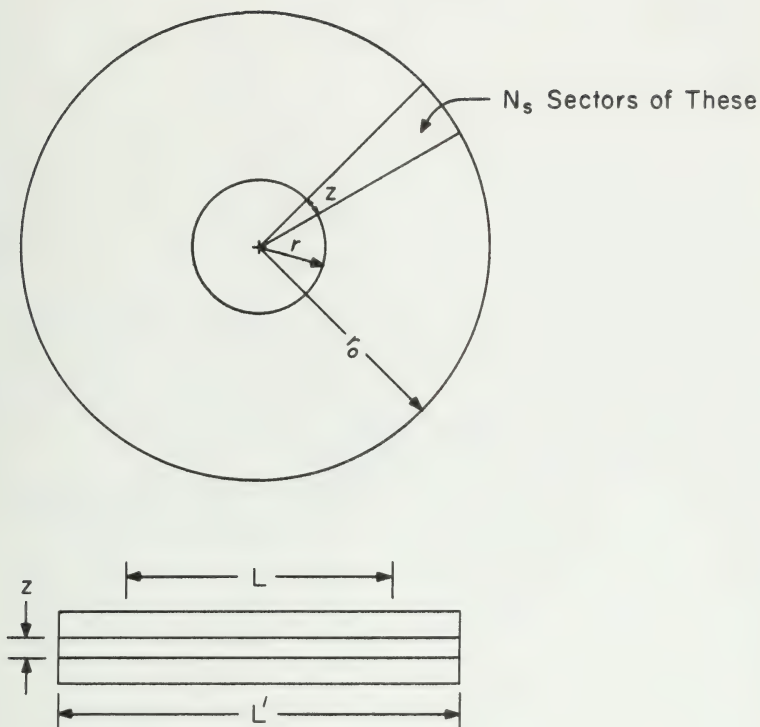


Figure 3. Encoding Disc and Line Detector

2.10.3 Principle of Operation

Assume that the tablet has equally spaced $2^k \times 2^k$ points. Every point source (x, y) on the tablet will have two unique line images, one for x and another for y . We can make an analogy between our device and the magnetic disc memory in the following sense. The line image with the line detector corresponds to the sense head when it moves radially in a disc memory. Our disc is an optical memory in this case, the words are arranged in tracks, each track represents one of the 2^k values of x (or y). During the rotation the line image with the line detector will read the bits recorded (in clear and dark lines) on the tracks. These bits could be any required encoding of the coordinates. If we confine ourselves to binary encoding of the coordinates we will get a serial binary number as output from the line detector.

Operation is as follows:

- 1) By pressing the tip of the light pen against the glass tablet, the pressure sensitive mechanism will turn the emitting diode on.
- 2) The radiated power from the diode will be transmitted and reflected by the semireflector.
- 3) Let us consider the encoding of one coordinate (y). The light incident on the y -lens is focused as a line of infrared light on the y -encoding disc.
- 4) The bit detector will detect the light transmitted through the encoding disc.
- 5) The output of the bit detector will be strobed by the clock and then synchronized by the sync. signal. The synchronized signal is fed to the y -register.
- 6) The y -register, upon request, will transmit the word bits to an output register.

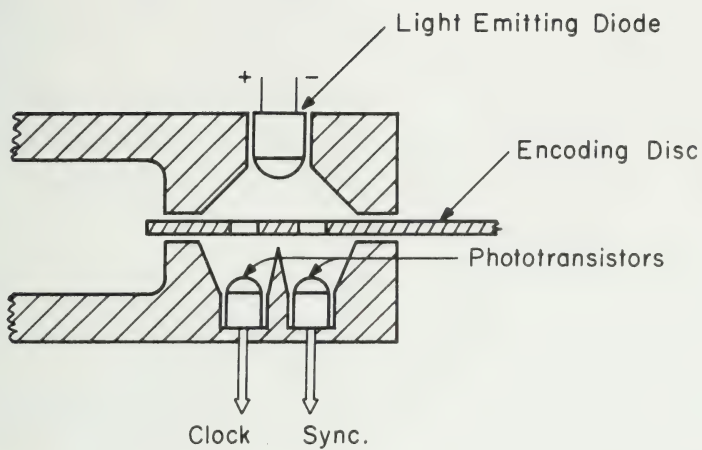


Figure 4. The Sense Head

2.10.4 Some Design Equations

The optimum design of the system depends on the light source characteristics as well as the dimensions and the relative positions of the different components.

Consider the case of y-encoding and let:

$P_1 \equiv$ The effective power incident on the bid detector in watts.

$Z \equiv$ The effective width of the line (bit) detector in meters.

$T \equiv$ The compound transmittance of the glass tablet, the semireflector, the lens, and the encoding disc.

$R_d(x, y) \equiv$ The detector spatial distribution when the pen is at position (x, y)

$R_i(x, y) \equiv$ The emitting diode spatial distribution when the pen is at position (x, y) .

$I_0 \equiv$ The maximum light intensity of the emitting diode ($x = y = 0$)

$D \equiv$ The width of the lens in meters

$f \equiv$ The focal length in meters

$d_0 \equiv$ The distance between the lens and the tablet in meters.

$d_i \equiv$ The distance between the lens and the encoding disc in meters.

$m \equiv$ The demagnification factor $= \frac{d_0}{d_i} = \frac{d_0 - f}{f} = \frac{f}{d_i - f}$.

In the following analysis we make use of simple geometrical optics.

It can be shown that

$$P_1 = TI_0 R_i(x, y) R_d(x, y) \cdot Z \cdot \frac{D}{f^2} \cdot F(m, x, y, f) \quad (1)$$

$$\text{Where } f(m, x, y, f) = \frac{m}{[(m+1)^2 + (\frac{m}{m+1} \cdot \frac{x}{f})^2 + (\frac{y}{f})^2]^{3/2}} \quad (2)$$

Let the tablet be a square with $A \times A$ meters and L be the effective length of the detector. We will have:

$$L \leq \frac{A}{m} \quad (3)$$

Therefore

$$m \leq \frac{A}{L}$$

A kind of compromise must be made between the various dimensions and the allowed distortion of the line image of the point source. This will influence the choice of the length of the detector L. This in turn gives us the allowed range of demagnification for certain A. The power incident on the detector can be maximized by the correct choice of the different parameters in Equation (1).

Now consider the design of the encoding disc (Figure 3) and let:

$R \equiv$ The resolution in lines/meter.

$V_w \equiv$ The linear writing speed in meter/sec.

$n \equiv$ The number of digits sensed/sec.

$Z \equiv$ The effective width of the detector.

$L \equiv$ The effective length of the detector.

$N \equiv$ Rotation speed in RPM.

$r \equiv$ The inner radius of the encoding disc.

Now

$$n = \frac{2\pi r_i}{Z} \cdot \frac{N}{60} \quad \text{digits/sec}$$

where $\frac{n}{k} =$ number of lines/sec. But $\frac{n}{k}$ must be greater than the number of lines scanned by the pen/sec.

$$\text{Therefore } \frac{n}{k} > RV_w$$

$$\text{Therefore } r_i > \frac{ZkRV_w}{N} \cdot \frac{30}{\pi}$$

$$\text{The total number of sectors } \equiv N_s = \frac{2\pi r}{Z}$$

$$\text{Therefore } N_s > \frac{60kRV_w}{N}$$

The outer radius of the encoding disc $\equiv r_0$ and thus

$$r_0 > r + L' + \Delta$$

where L' is the total length of the detector and Δ is the radial space required for the clocking and synchronizing holes.

2.10.5 Development

The above description is a new design of OPTITRACK. The previous design has been abandon because it was to expensive. Fiber optics were considered in that design. In addition to the cost of the fiber optics, the mechanical alignment was extremely critical. In the present design we make use of a special photodetector - namely, a line detector. This line detector senses the light incident on any region of its effective detecting area.

Mohmed El-Sonni

2.11 OCOMO (Optical Correlation Modulation, Project No. 42)

2.11.1 Introduction

OCOMO is a communication system employing light to encode and decode signals whose bandwidth is limited to 7.5KHZ such as audio signals.

While standard communication systems have used amplitude, frequency or phase modulation (and combinations thereof), OCOMO will use a pulse code modulation scheme. Figure 1 shows a simplified version of the OCOMO encoder. The heart of the encoder is a constant beam CRT with 64 film strips on the face of the tube. These film strips consist of transparent (1 level) and opaque (0 level) rectangles. Each code word is stored by constructing a sequence of transparent and opaque rectangles on each film strip. The code words are selected by first sampling an analog signal voltage. This voltage is then quantized and applied to the vertical deflection plates of the CRT. The code word is generated by applying a linear sweep of 64 μ sec duration to the horizontal deflection plates. The light signal is then converted to an electrical signal by a photocell. This electrical signal can then be transmitted.

Figure 2 shows a simplified version of the OCOMO decoder. The heart of the decoder is also a CRT with 64 film strips on its face. The horizontal

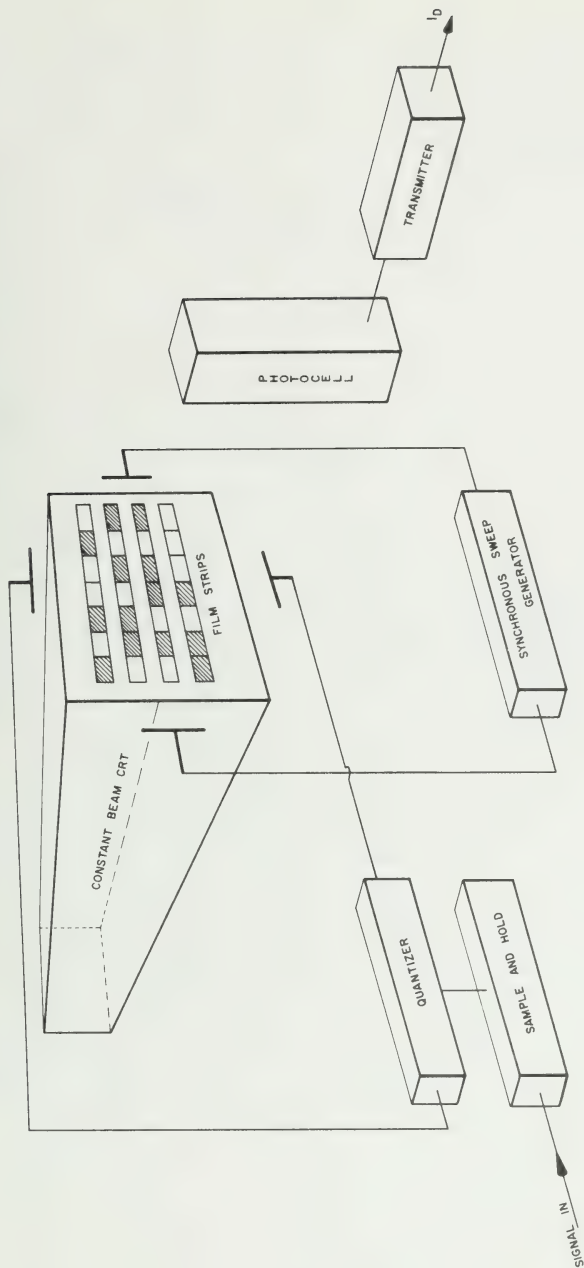


Figure 1. OCOMO Encoder

deflection plates are connected to a sweep generator which is synchronized to the encoder's sweep generator. The vertical deflection plates are connected to a ramp generator which deflects the CRT beam to every one of the 64 film strips during one bit of a code word. The CRT beam is modulated by the transmitted signal from the encoder. On the face of the CRT are the complements (opaque to transparent and vice-versa) to the 64 film strips at the encoder. The light from each of the film strips is transmitted to its own individual photocell using lenses or light pipes. When a code word is transmitted, the modulated CRT beam will be deflected to each strip once during each bit. Since the complements of the film strips are used, the film strip at the decoder corresponding to the code word sent will be dark during the entire code word and every other film strip will pass light at least once during the duration of the code word. Thus, if each of the 64 photocells are used to charge a capacitor, the photocell corresponding to the the transmitted word will not change its capacitor voltage. By determining which capacitor voltage has changed least, one determines the code word sent. Associated with each code word is the voltage corresponding to the encoder voltage.

The OCOMO system has many interesting features. Since there are 64 code words which can be arranged to correspond to 64 analog voltages in any order, there are $64!$ or about 10^{89} ways of arranging the correspondence. Another feature is the ease in changing the code word arrangement. Because the code is stored on film strips, one merely has to rearrange the strips in order to change the code. Also, provision can be made for transmitting the arrangement of code words from the encoder to the decoder by using a photochromic material at the decoder to record the code.

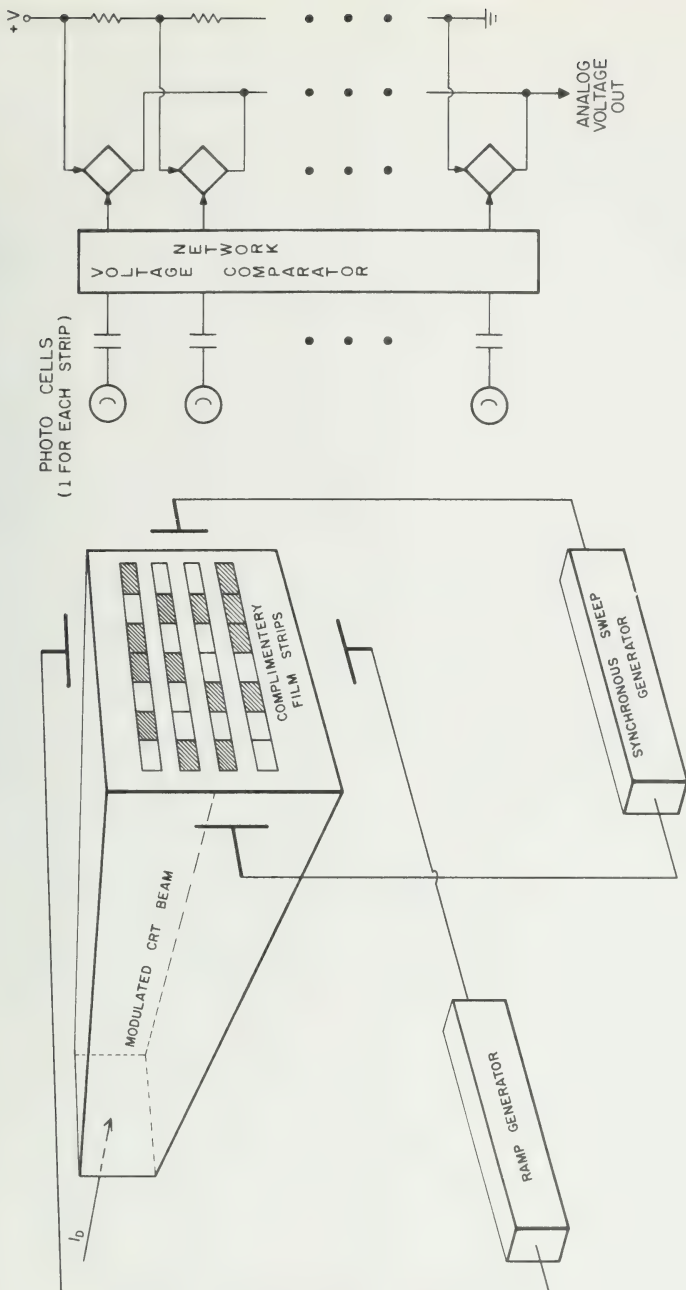


Figure 2. OCOMO Decoder

2.11.2 Present Work

The work has progressed well on the OCOMO encoder. An oscilloscope with a P16 phosphor (short persistence) has been ordered. Because the P16 phosphor has a spectral peak in the near ultra-violet region of the spectrum, a photomultiplier tube was chosen to detect the CRT light. Semiconductors have poor response in this wavelength region. The basic design for the vertical deflection system has been completed. The system consists of a sample and hold module, an A/D converter for quantization, a group of flip-flops for digital storage and a D/A converter to deflect the CRT beam. This design was chosen since it allows more than 60μsecs between samples which can be divided up between the acquisition time of the sample and hold module and the settling time of the A/D converter.

Some decisions have also been made about the OCOMO decoder. An oscilloscope with a P24 phosphor has been ordered. The P24 phosphor was chosen because its spectral peak wavelength is at 5000Å making the use of semiconductor photocells practical. The persistence of the P24 phosphor is longer than the P16 (500ns compared to a 50ns falltime), but this presents no problem since the shortest possible time between light activation of the same photocell is about 8μsec. Since each of the 64 film strips must have the beam deflected past them in 8μsecs, photodetectors with nanosecond risetimes are required. This short risetime necessitates the use of photodiodes. The vertical deflection system has been conceived. It will be a staircase generator built with a D/A converter and a counter.

A careful look at the codes revealed that a special code was required. A problem arises since an opaque spot and no CRT beam physically coincide. For example, consider two film strips which differ in only one bit position. Let film strip A have a 1 in the differing position. If the code word sent is associated with film strip A, the differing position at the

decoder will be a 0 (no light). Thus, no light will come from the differing position of either strip. Since the film strips are the same in all other positions, both film strips will pass no light. Thus, the decoder must decide which of two code words was sent. The problem can be solved by using hardware, but this solution requires each arrangement of the film strips to have the same ambiguities and, therefore, eliminates many combinations. Another solution to the problem would be to devise a set of code words which have no ambiguities. Such a code has been devised. This set of code words has the property that if one considers all of the zero positions in the selected word, all other code words of the code have a one in at least one of the positions corresponding to the zeros positions in the selected code word. Since complimenting filters are used at the decoder, a 0 at the selected film strip would block light but every other film strip would pass light in at least one of the positions where the selected word blocks light thus eliminating any ambiguity.

Robert Budzinski

PUBLICATIONS

- Donald Hanson, "Outlining and Shading Generation for a Color Television Display", Department of Computer Science Report 512, University of Illinois, Urbana-Champaign, June 1972, (Master's thesis)
- Takehiko Kato, "RASER: RANdom to SERIAL Converter", Department of Computer Science Report No. 515, University of Illinois, Urbana-Champaign, June 1972, (Ph.D. thesis)
- Charles Pirnat, "Observer Position Detector for the Stereomatrix 3-D Display System", Department of Computer Science Report 491, University of Illinois, Urbana-Champaign, February 1972, (Ph.D. thesis)
- Bernard Tse, "Supplementary Information on VISTA", Department of Computer Science Report 524, University of Illinois, Urbana-Champaign, June 1972
- Lawrence Wallman, "BLAST: A Stereoscopic Television Display", Department of Computer Science Report 511, University of Illinois, Urbana-Champaign, June 1972 (Ph.D. thesis)

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.
COO 1469-0211

2. TITLE
Quarterly Report - April, May, June

3. TYPE OF DOCUMENT (Check one):

☒ a. Scientific and technical report

☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

☒ a. AEC's normal announcement and distribution procedures may be followed.

☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.

☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

W. J. Poppelbaum
Principal Investigator

W. J. Kubitz
Assistant Professor of Computer Science

Organization

University of Illinois
Department of Computer Science
Urbana, Illinois

Signature

W. J. Poppelbaum

Date

July 1972

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

☐ a. AEC patent clearance has been granted by responsible AEC patent group.

☐ b. Report has been sent to responsible AEC patent group for clearance.

☐ c. Patent clearance not required.

3.1 Numerical Processes3.1.1 DIFSUB (R. L. Brown)

The problem of simulating a three-phase multiple-stack variable reluctance step motor was considered. It can be described by the following system of equations.

$$i'_a = [V_a - i_a r_a + ZL_1 \sin(z\theta + \frac{2}{3}\pi) i_a \theta'] / (L_0 + L_1 \cos(z\theta + \frac{2}{3}\pi))$$

$$i'_b = [V_b - i_b r_b + ZL_1 \sin(z\theta) i_b \theta'] / (L_0 + L_1 \cos(z\theta))$$

$$i'_c = [V_c - i_c r_c + ZL_1 \sin(z\theta - \frac{2}{3}\pi) i_c \theta'] / (L_0 + L_1 \cos(z\theta - \frac{2}{3}\pi))$$

$$\theta' = \omega$$

$$\omega' = (T - B\omega - T_f) / J$$

$$T = -\frac{ZL_1}{2} [i_a^2 \sin(z\theta + \frac{2}{3}\pi) + i_b^2 \sin(z\theta) + i_c^2 \sin(z\theta - \frac{2}{3}\pi)]$$

where for $n = a, b, c$

i_n is the current in the winding,

V_n is the "driving voltage" or potential across the winding,

r_n is the resistance in the winding,

T is the torque of the rotor,

θ is the angular displacement of the rotor,

ω is the angular velocity of the rotor,

and all other symbols are constants. A value of $Z = 16$ means that a change of phase occurs every 7.5° causing a change from 50V to 0V in one "driving voltage," an opposite change in the next voltage, and no change in the third.

The problem, which has been solved using a variable step size, fourth order Runge-Kutta integrator, was solved by DIFSUB using both the

stiff method and Adams method with only minor changes to the differentiation subroutine which worked with the RK integrator. The difficulty in using these multistep methods lies in the discontinuity of the "driving voltages," which change at specific intervals of angular displacement rather than at specific times, so no exact method exists to predict when these changes will occur and to adjust the step size accordingly. Thus, the program encounters difficulty in proceeding beyond these points using results computed from steps previous to encountering the discontinuity. The automatic step size and order changing routine in DIFSUB causes the program to either reduce the step size drastically while remaining at third order or else decrease the step size while decreasing the order to two or one. Either way, once the discontinuity has been passed, the program continues to use a small step size, increasing it more slowly than necessary since the solution is a fairly smooth curve except at the difficult points.

Nevertheless, DIFSUB, using the Adams method ($MF = 0$), correctly integrated the problem over the interval 0 to 0.1 sec. at a significantly reduced execution time; both DIFSUB methods used--Adams, and stiff with numerical evaluation of the matrix PW ($MF = 2$)--made significantly fewer calls to the differentiating subroutine DIFFUN. See Figure 1.

A possible improvement in performance is being investigated using a version of DIFSUB with a self-starting routine which takes one RK step and then proceeds at third order of the method chosen. If a way to recognize that a problem has reached a point where some variable (not necessarily one of the variables described by a differential equation) is discontinuous, then the program can be restarted at that point without having to build up from a small step size. Experience indicates that such a

recognition algorithm is possible and this is being investigated. An interactive simulation of the problem is also being considered.

Method	NFNS	TIME
RK	19,945	21.9
ADAMS	5,080	17.6
STIFF	12,366	38.0

Figure 1. Comparison of Three Step Motor Simulations

NFNS = Number of calls to differentiating subroutine

TIME = Seconds of actual 360/75 execution time of
compiled program

3.1.2 Sparse Eigen Problem (J. S. Deogun)

The sparse eigen problem was completed this quarter. Final stages of its completion were marked by the addition of the QZ subroutine by Moler and Stewart. A discussion of the basic procedure may be outlined as follows.

This program solves the sparse eigenvalue problem

$$Ax = \lambda Bx$$

A and B being general sparse matrices of rank considerably lower than their order. The idea is to reduce the sparse eigen problem $Ax = \lambda Bx$ to a non-sparse eigen problem

$$A1 X1 = \lambda B1 X1$$

where order of A1 and B1 is equal to the rank of B. The resulting system can now be solved by standard nonsparse techniques. The method is based upon progressive reduction by two-sided equivalence transformations of the original problem. The two-sided equivalence transformations as suggested by Wilkinson

can be affected by a cyclic process using Gaussian elimination. Each cycle consists of two steps.

In the first step, matrix B is reduced to upper trapezoidal form. To preserve the eigenvalues, similar operations are done on matrix A. Figure 2 below shows the form of matrix B--of order 6 and rank 4--after the first step. Here the upper left hand corner of B corresponds to matrix B1. In the figure, R_B stands for the rank of B.

$$\left[\begin{array}{cccc|cc} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right] \left. \vphantom{\begin{array}{c} x \\ 0 \\ 0 \\ 0 \end{array}} \right\} R_B$$

Figure 2

The second step consists of partitioning matrix A conformably with matrix B and performing Gaussian elimination on that part of matrix A which does not correspond to the upper trapezoidal part of matrix B, interchanging the conventional role of rows and columns and starting with the last row and column instead of the first. This is actually performed by restricting the choice of pivots to only those rows which do not correspond to pivoted rows in B and working on the transpose of matrix A. Of course, similar operations are done on matrix B. Figure 3 shows the form of matrix A after the second step. The upper left hand corner of matrix A corresponds to matrix A1.

$$\left[\begin{array}{cccc|cc} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ \hline 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{array} \right] \left. \vphantom{\begin{array}{c} x \\ x \\ x \\ x \end{array}} \right\} R_B$$

Figure 3

In fact, the progressive reduction by two-sided equivalence transformations is a cyclic progress, but in this case--the matrices A and B being of sparse nature--the reduction is completed in only one cycle. Then the reduced matrices A1 and B1 are changed to the usual array form from their sparse representation. Now, Moler and Stewart's "QZ" algorithm is employed to solve the resultant system

$$A1 X1 = \lambda B1 X1$$

The fact that this program produces results of remarkable accuracy was observed by running small handwritten examples with known eigenvalues. So far, only examples with square matrices A and B have been tested with the program, but all the functional routines are so designed that it requires little or no work to run the program with rectangular matrices A and B.

3.1.3 Auto-Restart (B. van Melle)

Action has been taken this quarter to implement an auto-restart facility in the numerical package. It has been found that in the process of solving for the initial conditions (DIFMF3), the values in the Y and YL arrays may vary so much that the matrix inverter generated by SPARSE uses

zero or near-zero pivots. In such a case, we would like to go back and have SPARSE generate a new matrix inverter, using the current values to guide pivot selection, and then return to roughly where we left off.

Since poor pivots usually manifest themselves in the form of floating-point overflow or zero-divide, a routine SPIEXT has been written to process such interrupts. SPIEXT fixes up the result register of the operation. It searches SPARSE's map to find out if the interrupt occurred in one of the generated routines, and, if so, in which sequence of operations the error occurred. Then a return code is issued indicating an auto-restart. If the interrupt occurred elsewhere, SPIEXT allows execution to continue after printing an appropriate error message. A separate entry, SPIEX2, processes calls to the FORTRAN error monitor (e.g. too large argument passed to DEXP) in a similar manner, except that such a call never originates in one of the routines generated by SPARSE. And a third entry, KNTSPI, called as a FORTRAN function, allows a program to access the error count.

The package has been written and is currently undergoing tests.

3.1.4 Plot Package (W. Chung)

The plot package has been running successfully during the last quarter period. Several modifications have been made to improve the structure and interactive behavior of the package.

1. Changes in command list:

Options for command mode are added to the old list to improve the dynamic nature of the display and to provide a kind of space-multiplying of the screen: <move> is to move the origin of the coordinate and draw the

curves whose scale is determined by the position of the origin; <scale> is to enlarge or shrink the picture in both directions (x and y).

Option <identify> is to identify a possibly unknown curve due to the bad labeling and that curve is specified by a graphical input using joystick.

Option <refer> for tutorial display was deleted because it needed too much core and the display itself was not very readable. Instead, for each delayed option which needs extra input, a short one-line guide is displayed.

Also, the order of commands in the menu was rearranged for simpler implementation of program switches and recovery from system or user errors.

The present commands are

- a. control commands--RECOVER(1), PLOTMODE(4), RESTART(5), QUIT(6)
- b. data commands--CLEAR(2), DISPLAY(3), PRINT(7), EVALUATE(8),
SHIFT UP(9), SHIFT DN(10), MOVE(11), ATTACH(12),
SCALE(13), IDENTIFY(14), LIST(15), DELETE(16),
SAVE(17)

Numbers in the parentheses represent the order in the menu.

2. Change in interaction for plot mode:

Now the user is given more freedom in specifying names of the variables to be plotted because the interaction is changed from question-answering which is program controlled to the user-driven methods. The names of saved variables and a few options are displayed in the menu:

REINIT--all parameters are reinitiated

TUTOR--tutorial display of four lines

IPNT--for specification of # of points

PLOT--let go!

Users can pick any or none of the variable names and/or options in arbitrary order. Any previously specified parameters can be changed easily by REINIT or picking IPNT again. PLOT should be selected as the last option. Default values are used for any nonspecified parameters.

As a result, a uniform and dynamic means of interaction is provided by joystick input only. Furthermore, the size and complexity of the program is reduced by shorter code for this section and by removing useless arrays IPLOT(15) and NAME(2) and parameters whereby the distinction between Y and YL variables has been made.

3. Labeling algorithm:

A new labeling algorithm is implemented based on the following assumptions and goals. First, labels are placed at the points where the distances between the curves are maximum. Second, the algorithm should not give rise to too much overhead in code and computation time.

(L1) Find IMAX such that $|YSV(*,IMAX)| = YPEAK$

If IMAX = 1, then IM = IMAX and IMAX = I such that

$|YSV(*,I)/YSV(*,IM)| \leq 0.95$. Go to (L2).

Otherwise, find IM such that $YSV(*,IM)/YSV(*,IMAX) \leq .95$

(L2) Sort the array INDEX(*) in the order of YSV(INDEX(*),IMAX) value. I = 1.

(L3) If $I \geq IPT-1$, then go to (L6). Otherwise I = I+1;

If $|YSV(INDEX(I),IMAX) - YSV(INDEX(I-1),IMAX)| \geq 0.05 * YPEAK$,
then IL = IMAX and go to (L4);

If $|YSV(INDEX(I),IMAX)-YSV(INDEX(I+1),IMAX)| \geq 0.05*YPEAK$,
 then IL = IMAX and go to (L5);

If $|YSV(INDEX(I),IM)-YSV(INDEX(I-1),IM)| \geq 0.05*YPEAK$,
 then IL = IM go to (L4);

If $|YSV(INDEX(I),IM)-YSV(INDEX(I+1),IM)| \geq 0.05*YPEAK$,
 then IL = IM and go to (L5). Otherwise, put the label
 at the end of the curve and repeat (L3).

(L4) Put the label above YSV(INDEX(I),IL) and go to (L3).

(L5) Put the label below YSV(INDEX(I),IL) and go to (L3).

(L6) Put the labels above YSV(INDEX(1),IMAX) and below
 YSV(INDEX(IPT),IMAX). The algorithm terminates.

Though the conflicting requirements for the visibility of labels and the efficiency of algorithm is hard to meet, the present algorithm improves the readability quite a bit for current examples.

4. Plans for the next quarter:

- a. Improvement of labeling algorithm.
- b. Improvement of user-program interaction, especially for the save mode.
- c. Feedback in the design process will be improved through interactive changing of design parameters. For this purpose another set of plots which is a display of one variable for various parameter specifications will be added to the present plot of several variables for a particular design specification.

Problems to be solved are to define the "best" design, to devise an iterative algorithm changing design parameters,

to define the form of graphical input and output, to implement the data structure which will be adequate for saving and analyzing the new set of plots.

- d. Dynamic user-program interaction and mathematical analysis for the smoother and accurate display of curves. So far, the program has absolute control of how many and which data points should be saved for display, and the generated graph is a "polygonal line" which usually does not appear to the eye as a smooth curve.

The data points will be interactively determined by the user with maximum flexibility. The implementation for deletion and addition of data points which are returned from DIFSUB will be based on both visual and mathematical analysis with consideration of curvature and continuity. Also, the selection and computation of non-nodal points will be controlled by the user to produce "beautiful" graphs. Optimization in the sense of fast real-time response and program size should be a determining criterion.

- e. Hardcopy output of the above plots will be generated by the CalComp plotter for documentation and later analysis of the design.

3.1.5 Item Analysis (J. Koch)

Terminal type definition was removed from the picture data structure and made into a command interpreted by Item Analysis. If a terminal type definition is present in a picture, the message 'NO TERMINAL TYPE DEFINITION ALLOWED IN PICTURE.' is sent but the picture is still itemized. The new command is in the following format.

DEFINE type: E(variable, variable,...), I(variable,...). The message 'TERMINAL TYPE IS NOW DEFINED.' is sent after the type has been stored on NODLIB. The following error messages are sent if the DEFINE command is in improper format;

1. 'MISSING': ' IN TYPE DEFINITION.'
2. 'MISSING') ' IN TYPE DEFINITION.'
3. 'VARIABLE TYPE IN DEFINITION NOT 'E' OR 'I'.
4. 'NO E OR I VARIABLES SPECIFIED FOR TYPE.'
5. 'DEFINE' MUST BE FOLLOWED WITH A TYPE DEFINITION.'

Item Analysis checks the blocks present in a picture and prints the following message if the picture cannot be put through Item Analysis: 'PICTURE NOT 'ITEMIZABLE.' NO SUBPICS OR EQNS.'

FIOCS#

The routine to do FORTRAN I/O was modified and tested. Special cases for I/O to units 0, 1, 2, 5, 6 were included. A subroutine call, FI099Z, will now allow I/O to an in-core array instead of an I/O unit. The call to FI099Z has been made compatible with the version of FI099Z available currently to Illinois users. FIOCS# was also changed to allow input/output to nonstandard unit numbers. All DCB parameters for these units must be given on the DD card. No defaults are supplied. The following table indicates the special cases and what labels are branched to in FIOCS#:

	Open read	Open write	read	write
Unit 0	FINIT/BRITE SWRET			FRITE/WALL
Unit 1	FINIT BREAD SBREAD		FREAD/SBREAD	
Unit 2	FINIT/BRITE SWRET			FRITE/WCON WCOM
Unit 5	FINIT/BREAD		FREAD/BREAD1	
Unit 6	FINIT/BRITE WRET			FRITE/WRET
Unit 3-4, 7-99 0,2,6 on read 1,5 on write	FINIT/BREAD GR	FINIT/BRITE GW	FREAD/BRET	FRITE/WCON FGW/FGWW
FI099Z 3-4, 7-99 0,2,6 on read 1,5 on write	FINIT/BREAD GR/GW1	FINIT/BRITE GW/GW2	FREAD/FGW1	FRITE/WCON FGW/FGW1

3.1.6 FINDEX (J. Stynes)

General Background

Every model, or network, constructed by a user is composed of elements, which, in turn, may be composed of other elements. (There is no restriction on the level of nesting.) Associated with each element are specific variables, such as parameters, global and local variables, terminal variables (called E- or I-variables), etc. For example, the element, resistor, might have a parameter, R; terminal variables, V and I; and a global variable, TEMP, for temperature. Each user-named variable, such as R, V, I, or TEMP, has associated with it an internally generated system variable. The purpose of the program, FINDEX, is to find the internal variable associated with a given user-named variable. However, in order to find the correct variable, the element associated with this variable must

be unambiguously defined. To do this, the system's internal representation of a network must be understood.

To describe the various interdependencies of the elements in a particular network, the system uses a tree structure within which each tree node corresponds to a particular element in the network. These nodes are linked together using SON pointers and SIBLING pointers. Figure 4 is an example of a possible network, including a graphical representation of its tree structure. From the graph it is evident that element A, the top level element, is composed of elements B, C, D, and two elements of E. The SON pointer of element A points to element B, while B's SON pointer points to D, its SIBLING pointer points to E, and so on. The last sibling on any level points back up to the father on the level directly above. Thus, element C points back to A. If an element has no SON or SIBLING, its pointer is zero.

Also associated with each element are its terminal connection points. These are the points at which another element may be connected. For example, the elements B and D in the sample network each have two terminal connection points. When elements are connected together in a network, or subnetwork, their terminals are given a unique numeric value within that subnetwork. For example, the terminals on the element E have the values 2 and 3 in subnetworks D and B, while in network A one element E has terminal values 2 and 3 and the other has values 6 and 7. It should be apparent that the specification of the name of an element and one of its terminal numbers suffices to unambiguously define that element within a particular subnetwork. However, ambiguities might still arise because an element might have the same name and terminal numbers in two different subnetworks. In the sample

network, element E has the same name and terminal numbers in both subnetworks B and D. Therefore, in order to clearly define one of the E elements with respect to the whole network, additional levels of specification are needed. The method of specification used in FINDEX is as follows.

The user must uniquely identify an element as the top level element, or starting point. He must then mentally construct a path from this element to the element he wishes to identify, being careful to uniquely identify each element through which the path passes. That is, after choosing the top level, the user must specify an element on each level thereafter until the desired element is reached. The user does not necessarily have to draw the tree graph for his network. He must simply have an understanding of his network, and the possible ambiguities that are present. It is also helpful to realize that on a particular level an element may be uniquely defined solely in terms of its terminal number, and if an element only appears once on any level, it may be specified uniquely by either its name or its terminal number.

The calling sequence for FINDEX is:

CALL FINDEX(VAR,ELEM,PDPNO,LEVEL,I,J)

where:

Input: VAR: The name of a double-length word which contains the variable whose internal variable is desired.

 ELEM: A double-word array containing the elements needed to uniquely specify the element containing VAR. (If an element name is not needed on a particular level, its corresponding double word is blanked out.)

PDPNO: A single-word array containing the terminal numbers associated with each of the elements in ELEM. (If a PDPNO is not needed for a particular level, it is given the value -1. It must be remembered that either a terminal number or an element name, or both, must be specified for each level.)

LEVEL: The number of elements in ELEM. (The number of levels needed in defining the element containing VAR.)

Output: I: Number describing the type of internal variable, or type of error.

0 G variable

1 T variable

2 YL variable

3 Y variable

4 Y' variable

5 constant

-1 error in input

-2 inadequate information

J: + or - index into array

Sign is whether symbolic name is + or - the array element

Some examples using the sample network in Figure 4 follow. FORTRAN data statement formats are used to indicate the contents of each of the variables or arrays.

a. To find the internal variable for some terminal variable, X, on terminal 2 of the element E numbered ① in the figure:

```
CALL FINDEX(VAR,ELEM,PDPNO,2,I,J)
```

where

```
VAR = /'X'/'
```

```
ELEM = /'D 00000000', 'E 00000000'/'
```

```
PDPNO = /0,2/
```

A possible output would be

```
I = 2
```

```
J = 20
```

corresponding to YL(20)

Note: If a terminal variable is specified, a terminal number must be specified for that variable.

b. To find the internal variable for some local variable, Z, on element C, numbered 2 in the figure:

```
CALL FINDEX(VAR,ELEM,PDPNO,1,I,J)
```

where

```
VAR = /'Z'/'
```

```
ELEM = /'C 00000000'/'
```

```
PDPNO = /-1/
```

c. To find the internal variable for some parameter, P, on element E, numbered 3 in the figure:

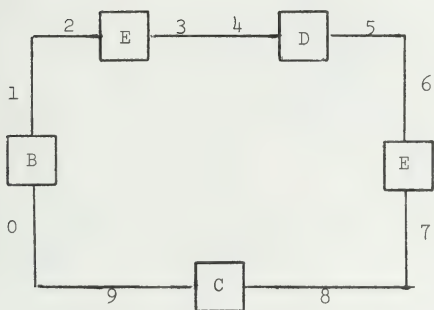
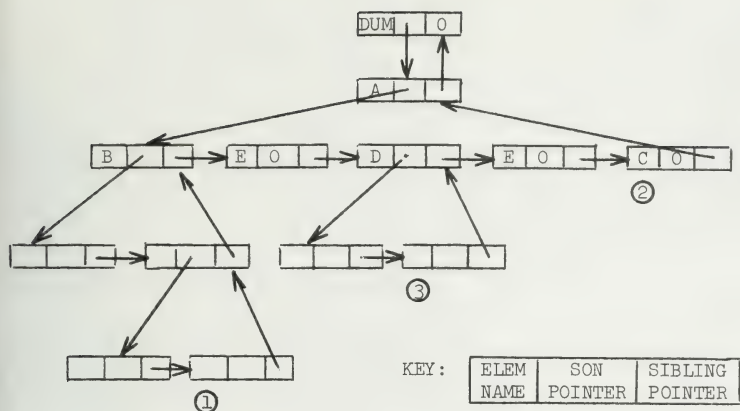
```
ELEM = /'D 00000000', ' 00000000'/'
```

```
PDPNO = /5,2/
```

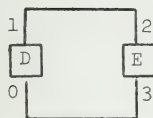
Short Description of the Program

The program is essentially in two parts. The first part is an algorithm to search the tree for the desired element, while the second part searches various tables for the internal variable associated with the given variable.

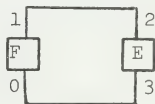
The flowchart in Figure 5 describes the algorithm used in searching the tree. Within the program itself, each node is represented by two DSECTS. The first is called TREEDS which contains, among other things, the SON and SIBLING pointers, the element name, and a pointer to the second DSECT, called DLIST1, which contains the terminal numbers of the element. (The pointers SON, SIBLING, and LIST1 are displacements from the address of the top of the tree contained in TREEPTR. In DLIST1 the displacement DISPTNT is the displacement from the address of the top of the Terminal Type Name Table contained in TERMPTR. The other displacements are calculated in a similar manner with other bases. It should also be noted that the first entry in the tree is a dummy entry which points to the top level of the tree.) After the element which contains VAR has been found, the second part of the program simply indexes into a number of different tables to find the appropriate interval variable. If it is found, its type is returned in I, its index into the particular array is calculated and returned in J. If not, an error message is returned in I.



Network A. (Network for above tree)



Subnetwork B (within network A)



Subnetwork D (within networks A & B)

Figure 4

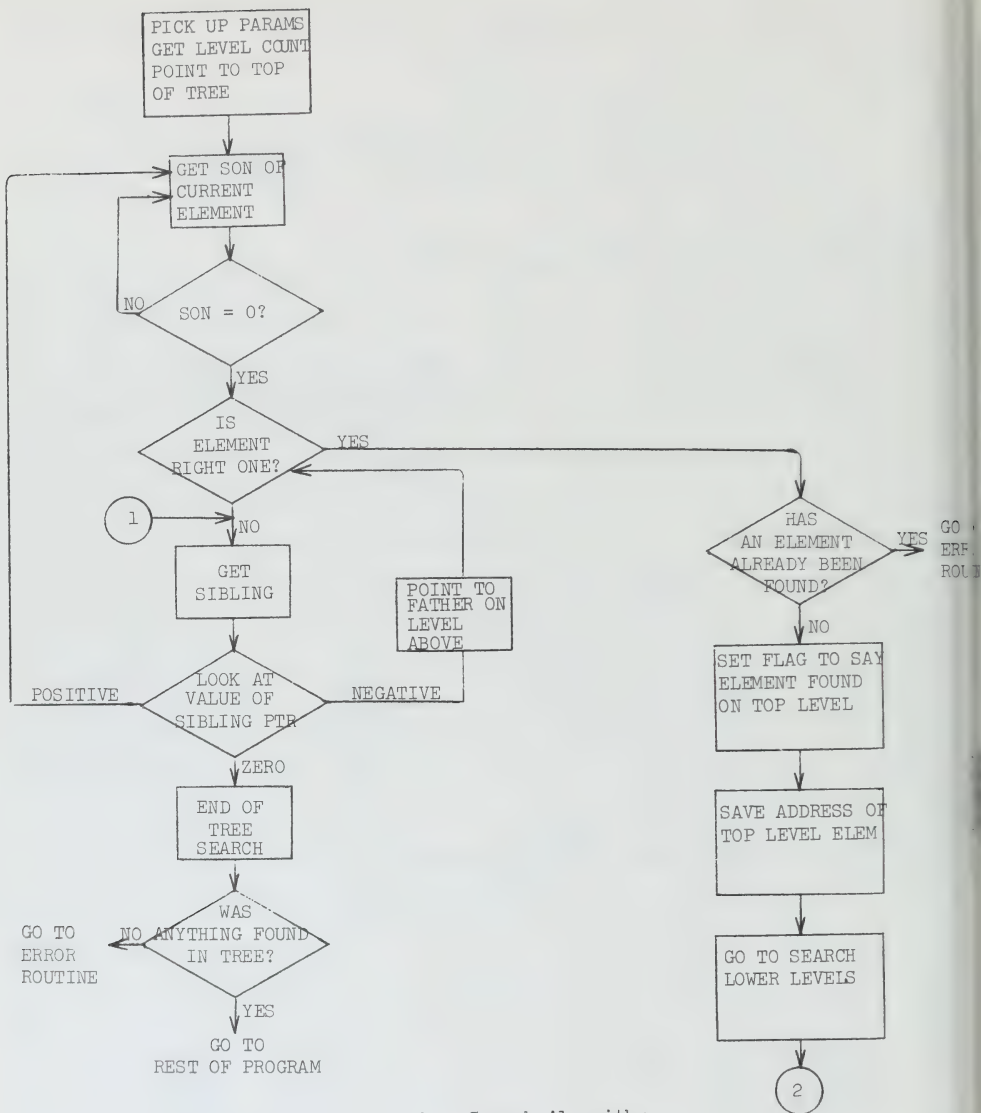


Figure 5. Tree Search Algorithm

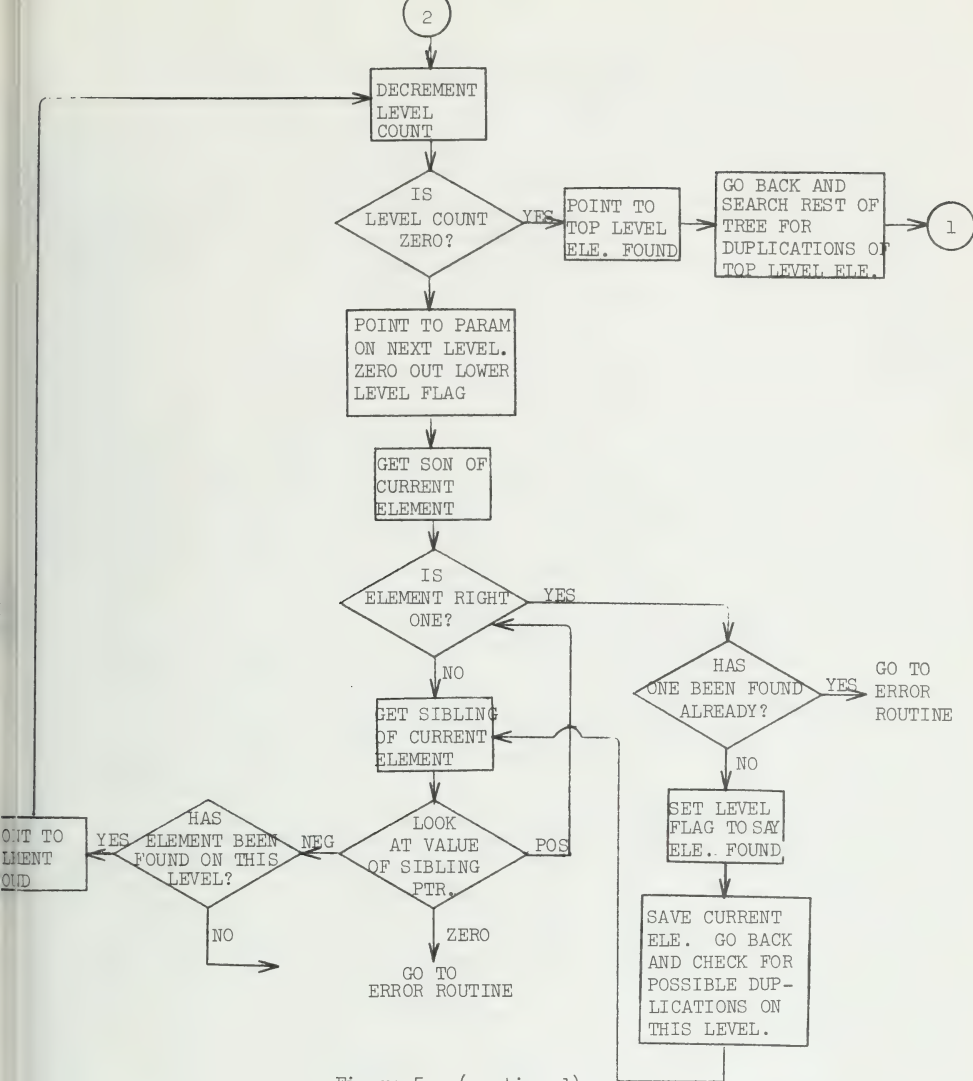


Figure 5. (continued)

3.2 Illinois Graphics Computing System (IGCS)

3.2.1 Overlay Supervisor (D. Mueller)

The overlay supervisor was completed during the spring of 1972. This package brings predefined modules into core and provides facilities for modules to overlay themselves. Overlays may be either on disk or DECtape, and facilities are provided for updating the supervisor tables should the location, length, or starting address of an overlay change. Two facilities are provided for initializing the monitor tables. One allocates dynamically and is intended for debugging purposes. The second has the tables predefined and is intended for production use. Programs are also provided to create overlays from load modules produced by the Linker.

3.2.2 A Drafting Language (ADL) (T. Runge)

Compilation of ADL jobs will be performed in two passes. The first of these passes reads an input card deck, lists it complete with diagnostics, interprets the specified figures and other functions (such as translation of the origin, rescaling, etc.), and produces an output file consisting of figure type codes, and figure parameters such as endpoint or center coordinates, radii, strings of ASCII text, arc sizes, etc. ILLISM will also produce an output file of this form and thus will use only the second pass of the compiler. This first pass was coded during the spring of 1972.

The second pass of the compiler performs two functions. The first of these is performed by the entry routines, which were coded prior to this quarter. They use ILLISM or first pass output to generate the data required for iterative plotting of the figures. The second function,

sectorization of the plot, is performed by the routine GETBUF, which was coded this quarter. GETBUF instructs the entry routines as to where to place the data they generate and performs the buffering of this data to disk. A partitioned file is thus created with data for all figures whose uppermost points fall within a particular interval of 30 scan lines sharing a partition.

3.2.3 Gould Printer/Plotter Interface (M. Clifton)

Supervisor for the third overlay: After all figure data has been preprocessed and stored, this routine is overlayed into core to control the creation of bit maps representing the drawing. The main functions of the program are controlling the input of figure data for each sector, examining the data and calling the plotting subroutines to create the bit maps, the storage of the completed bit maps, and storage of any figure data that extends beyond its immediate sector. After the figure data has been exhausted, the final step is to dump the completed bit maps to the Gould.

A program to graph specified values of a FORTRAN function subroutine on the Gould 4800 was begun. Some necessary specifications for the program are a start and end value for the x-axis, a start value for the y-axis, the ration of axis length per inch of plot and optionally some form of labeling and printing of the graph's value in an E-format.

A few changes were made in the line printer program for the Gould 4800. These were the resolution of the problem which caused the printer to halt while outputting a file, allowing output to be pages, and providing a small measure of line control for FORTRAN output files.

3.2.4 ILLISM-E (W. Tam)

The input section of ILLISM-E was completed. At the moment, only the teletype version is available. A batch version will be implemented depending on the availability of associated system software. The command language in ILLISM-E is keyword oriented. It will go into different modes by typing in the appropriate keyword and specifications. The keywords include

ELEMENT-- for defining elements. The syntax is '#ELEMNT name'

in which an element called 'name' will be defined. After the keyword is typed, ILLISM-E will go into the element definition mode, whereupon the specifications of the element will be typed in subsequently, also by using keywords.

NETWRK-- for synthesizing elements into networks, has the same syntax and function as ELEMNT, with a list of elements to be connected, and specifications on the connections.

RUN-- start simulation on a given network. ILLISM-E will request the unspecified parameters as well as the variables (nodes) to be monitored during the simulation.

EXAM-- a debugging tool to list any element or network for checking purposes

Current work is on the synthesis of elements into networks and the generation of executable code to interact with DIFSUB, the numerical analysis part of the system, as well as communication with the output package. All the codings are in a modular structure. This enables the system to be specified as a series of macros, and hence easily implemented on another small computer for greater portability.

3.2.5 Numerical Package (R. L. Brown)

DIF11, an O.D.E. solving program which uses a version of DIFSUB, is now running on the IGCS system's PDP-11/20. The program was prepared with an interface designed to be compatible with the ILLISM compiler, thus allowing its use in the development of the compiler.

The present version of DIF11 is written mainly in FORTRAN IV as compiled by the DEC version 3A FORTRAN compiler. The core-resident main program segment, written in assembly language, accepts the initial value of the independent variable T , the number N of variables in the system, the interval H over which the system is to be integrated, and the starting addresses of the $Y(I)$ and $DY(I)$ vectors, these addresses being set by ILLISM. The initial values are left in $Y(I)$ by the compiler. The main segment next allocates the addresses of the other vectors and arrays used by DIFSUB. The control program segment for DIFSUB is then called.

The control program segment executes one step using an Adams method version of DIFSUB, making extensive use of seven disk-resident overlays which are called into a single subroutine overlay buffer. The first such overlay is the differentiating subroutine output by ILLISM. The overlays are currently being called into core by the DEC-supplied LINK routine available with version 7A of the LINK editor. All communications between the control program segment and the overlays are through FORTRAN COMMON arrays (assembly language named .CSECT's) except for the number of variables N and the arrays whose addresses are set by the main program segment--these being passed as subroutine parameters by executing an intermediate assembly language routine in each overlay which places the

addresses in a FORTRAN subroutine calling sequence argument list and executes the overlay as a FORTRAN subroutine. The array addresses are not available to the link editor when the COMMON areas are created and, therefore, cannot be passed in that way; the overlays involve arrays whose dimensions are variable with N and FORTRAN requires N to be passed as a subroutine parameter in that case.

Output is done by a FORTRAN program section called by the main section on exiting from DIFSUB. Output is currently sent to the teletype, but future versions will output to DECTape for further analysis and possible graphic output.

The program has been tested extensively using common one-variable differential equations with known solutions and the results were compared with tables to verify their accuracy. Systems of 2, 3, 4, and 10 variables were also tested and were found to execute accurately. The present version executes with a delay of several seconds between steps, but this is due to LINK looking up the disk address of every overlay everytime it is requested.

Planned improvements include replacing LINK with a routine used with a previous version of DIFSUB which keeps a table of disk addresses. Also, a version of DIFSUB with an automatic step size initializer and an automatic change of method routine (between Adams and stiff) is being written. This version should replace the current version this fall. A stand-alone version of DIF11 (independent of ILLISM) is being considered. It would require the user to write, compile, and link a FORTRAN differentiating routine himself and use this as input to DIF11. This would allow IGCS users to solve O.D.E.'s before the whole ILLISM package is ready, and could be used for problems which ILLISM is not designed to handle.

3.3 Graphical Remote Access Support System

3.3.1 OS/8 Operating System (J. Nickolls, C. Segre)

The new OS/8 software arrived from DEC and has been successfully implemented on our PDP-8. The switchover from the Disk Monitor System to OS/8 will provide a much better tool for program development than did the DMS. A much improved Editor plus a cross-reference from the PAL8 assembler should speed program debugging and editing considerably.

Device handlers for our Systems Industries disk and Inktronic printer have been written and successfully inserted into the system. Programs are being written to convert files from DMS format to OS/8 format, and to allow the user of the disk as a system device. Also, a program to aid in system patching and modification is nearly completed, and will be fully documented in the next quarter.

3.3.2 Information Retrieval (J. Nickolls)

The new version of IR has been in production use with GRASS for nearly two quarters now. The new disk from Systems Industries has had no errors whatsoever, and IR has been totally reliable.

3.3.3 Monitors (J. Nickolls, R. Haskin, C. Segre)

GLASP (PDP-8 Local Monitor)

Work is continuing on a major revision of the local system which will handle the new joystick pushbutton functions and eliminate several problems inherent in the current system.

The library menu display routines are being redesigned, since the current method has proven disastrous on occasion. Previously, the menu was

kept in core and rewritten on the local disk only when the user logged out. Now, the menu will be created from the user's file directory at log on time, hence avoiding problems if the user forgets to log out or if the system goes down.

Prior to the installation of the new IR system, all files were common to all users, i.e. the user ID codes in the directory were all zero. At the time of the change of IR systems, the user ID's were placed in the directory, since it was a good time to do so. Unfortunately, there is presently no way to copy files from one user's library to another (except through the 360). The library routines are being changed to enable the Local Library Service Discographer (LLSD) program to copy another user's files. Also, file protection will be implemented at the same time.

It would be desirable for a GRASS user to be able to design and insert new functions and modes into the local PDP-8 system. To realize this, a high-level language is being investigated. This language would be compiled on the IBM 360, and the compiled code would be interpreted by the PDP-8 monitor. The source language is currently being designed.

Program Segments: GSAMV2

To speed use of the local modeling system, "SPECIFY" mode has been removed, and the six parameter specification functions are now in the initial program menu along with "CONSTR" and "REACT." A new keyboard function was also added which allows transfers from one GSAMV1 submode to another. It is of the form: !!NAME where NAME is the name of the submode to which transfer is desired. The name may be abbreviated to just the first two letters, i.e. !!EQ = !!EQTNS.

3.4 Computer Maintenance and Construction

3.4.1 Graphics-8 Hardware (C. E. Carter)

Data Disk

After arrival of new Systems Industries disk and controller, contacts were renewed with Data Disc in an effort to have the disk examined in California. Data Disc agreed to do this work at no cost to us. So, the disk is now in their shop being examined for troubles associated with the last work they were supposed to have done.

Systems Industries Disk and Controller

The new disk was installed, checked-out, and put on warranty this quarter. As far as can be determined, it has operated correctly since its initial check-out.

On-Going Projects

1. The tables for the new graphic consoles are ready for assembly.
This will make all terminals alike.
2. The second new design joystick is being checked-out and a third mechanical portion is on order.
3. The interface for the additional 4k of memory is ready for installation. This will make 20k of core on the PDP-8.
4. New and better cards for the Computek are coming out of the shop and will be used in a new terminal.
5. The character generators on all graphic terminals have been changed to include an (f) integral sign.

6. A crystal-controlled clock is now serving the console teletype on the PDP-8.
7. Systems Industries controller interface drawings are complete.

3.4.2 Equipment Maintenance Log Summary (H. Lopeman, R. Miller)

PDP-8

1. Teletype problems with the teletype interface, clock drifting. Installed crystal clock modification--problem corrected.
2. Indicator lights burned out in control panel. Replaced.

PDP-8/8I Channel

1. Channel problems of losing flags, garbage on computeks, etc. Found bad load gate on laser interface (Cunningham). This gated garbage into the 8/I accumulator periodically.

ComputeK

- Term #1:
1. Joystick box buttons bouncing. Characters garbled at times. Cleaned up pushbutton filtering system.
 2. Bad characters received and displayed. Cleaned contacts on multiplexor board for Term #1.
- Term #0:
1. Erase problem on 611 storage scope. Replaced driver transistors in erase and collimation circuitry.
 2. Function button panel loose. Replaced.

Inktronic

1. Paper feed drive belt broken. Over-run clutch seizing up.
Lubricated and freed. New drive belts ordered.

PDP-7 and 630 Maintenance

1. Replaced reader lamp, open filament.
 2. Installed circuits in 630 for 30 character terminal port #40 was used for this terminal.
 3. Installed four switches in 630 so that ports 40 & 03 can be switched to either 30 or 10 character terminal use.
 4. Check-out and testing of ports 40 & 03 circuitry for 30 character and 10 terminals.
 5. Replaced PCB's B11 & A9 concerning failure to read 7 and 8 hole combinations.
 6. Checkerboard test failed to run correctly, found bump diode in PCB 003 shorted in AC bit 12.
 7. Cleaned all air filters in PDP-7 and 630.
 8. Replaced 4706 rec. board in port 50, also 4707 trans. in port 33 & 21.
 9. Speed control knob on PDP-7 had stripped threads, retreaded knob and replaced screw.
- Driver transistor in bits 16 and 17 in AC register need replacement.

PUBLICATIONS

Cunningham, Ian "Hardware/Software Interface for the Stereomatrix Display," Department of Computer Science Report UIUCDCS-R-72-508, Urbana, Illinois 61801, June 1972; submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science, June 1972.

van Melle, Bill "AN OPERATING MANUAL FOR DIFMF3: A Program to Solve Initial Conditions in Simultaneous Differential Equations," Department of Computer Science File UIUCDCS-F-72-870, Urbana, Illinois 61801, April 1972.

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

COO-1469-0211

2. TITLE

2nd QUARTERLY REPORT (April, May, June) 1972

3. TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:
 Title of conference _____
 Date of conference _____
 Exact location of conference _____
 Sponsoring organization _____
☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

C. William Gear, Professor
and Principal Investigator

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature

Charles W. Gear

Date

June 30, 1972

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

4. IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH: ILLIAC III
(Supported in part by Contract AT(11-1)-2118 with the U.S. Atomic
Energy Commission)

4.1. PRINCIPAL DEVELOPMENTS IN BRIEF

A program of image processing experiments is being carried out. Our goal is to design a sight sensory system predicated upon parallel strategies for image processing. This development must, of course, evolve from our experience with the Illiac III system. The key property we seek is the ability of the system to learn rapidly from instructional interaction with professional personnel (not necessarily trained in computer technology). The validity of this orientation has been, we believe, vindicated by the rapid growth in significant image processing applications we can now attach.

Highlights of the program this past quarter include:

A. Interactive Picture Processing

1. Show-and-Tell has bedded down to being a workable, documented package for interactive picture processing.
2. An Atlas Extension of Show-and-Tell, described below, has emerged from the planning and early implementation stage toward a viable strategy for the effective use of map-defined information in the interpretation of images.

B. Pattern Recognition

1. Covering Theory concept--extending to pattern recognition methods originating in switching theory, continues to be a very fertile ground for investigation. Interval Coverings have previously been described in our internal reports. Now Cartesian Covers, the latest extension of the covering theory methodology, have been introduced--accompanied by an operational program for their construction.

2. Vari-valued Logic, an outgrowth of the covering theory work which extrapolates from two-valued variables of switching theory to quantized variables, has been introduced to provide a decision calculus for dealing with the pattern recognition problem.

C. Scene Analysis

1. Texture Analysis. A successful merger of concepts from interval covering theory and signal detection theory shows every promise of providing a first successful attack upon the problem of texture discrimination and analysis. Texture and color provide essential attributes for scene segmentation, i.e., the identification in the scene of candidate objects and their associated regions. Texture analysis is now generally recognized as a key obstacle to widespread application of image processing.
2. Shape Recognition, a necessary ingredient of the ATLAS system, has been successfully attacked in the work of Maruyama for simple two-dimensional shapes called "angularly simple." Extension to more involved forms is proceeding apace.
3. The ATLAS software package, an eidetic-based system for the processing of visual data, is emerging from the cocoon of its initial conception. Attempted applications to brain mapping and to aerial photointerpretation have stimulated much examination and extension of these concepts. A mathematical statement of the problem, how to use map information to interpret visual imagery, has been formulated.

D. Applications

1. Automated Cervical Smear Analysis

The prime objective was to examine the effectiveness of a parallel

digital image processor in the analysis of cervical smear imagery. The emphasis here was not on the discovery and extraction of parameters of a malignant cell from a nonmalignant cell, but rather on the construction of algorithms for a parallel processor which permits the computer to rapidly make sense out of the mess of cells and debris in the microscope field so that subsequent measurements (whatever they might be) are made on the correct objects: epithelial cell nuclei, for example, rather than clumps of white blood cells, cytoplasmic folds or locally similar-appearing phenomena.

A corollary investigation attempts to characterize the chromatin patterning of the nuclei, based on the texture analysis procedures mentioned above.

2. Brain Mapping. A feasibility study of the automatic scanning of cell nuclei in brain tissue is being undertaken. Shape analysis of the nuclei is used to quantify transneuronal effects: cell dilation, cell attrification, etc. The tissue section is scanned under control of the automated light microscope, with digital control of the stage and focus.
3. Multi-Spectral Analysis. Imagery, both biological and remote sensing, where color plays a strong discriminatory role, are being examined by the prototype mechanism of using three-color separation negatives. Texture analysis techniques are being extended to these additional local (i.e., chromatic) attributes.

4.2. INTERACTIVE PICTURE PROCESSING

4.2.1. GENERAL ORIENTATION

Facilities were proposed last year which would permit personnel (not trained in computer technology) to input scanning instructions directly to the image processing system. These included:

- (1) The interactive Show-and-Tell Programming System which would allow rapid development and evaluation of image processing algorithms,
- (2) The Atlas extension to Show-and-Tell which was to permit one to build in rapidly and conveniently predefined structural knowledge of the visual scene, and
- (3) Scene segmentation strategies and structural inference procedures which should allow the system, under minimal guidance, to rapidly infer which factors of the scene best articulate the given visual environment.

The first goal, Show-and-Tell, has been fully realized (e.f. 4.2.2.1) with the system undergoing minor, continuous improvement. The second goal, the Atlas Extension, has entailed the development of the Structure Operation Language (SOL) (e.f. 4.2.2.3) and extensive investigation of image deformation (to establish correspondence between map information and image information) (4.4.4), shape identification (4.4.5), and structure transformations (4.4.6). These parts have not yet been assembled into an operable system, though the thesis of Maruyama provides a first comprehensive trial.

The third ingredient, Scene Segmentation and Structural Inference, is finding its proper interpretation in the pattern recognition work, Section 4.2.3. Here covering theory and signal detection theory are providing adequate, if yet little understood, tools.

4.2.2 PICTURE PROCESSING LANGUAGES AND THEIR IMPLEMENTATION

The development of picture processing languages is considered a necessary part of interactive image analysis. Currently, adequate languages exist to express algorithms which operate on the array representation of pictures as binary valued elements with neighborhood connectivity relationships. The next and most natural abstraction from the array representation is a graph structure, by means of which many scene segmentation algorithms can be simply expressed. Our strategy here has been to experiment and develop theory for the level of picture processing operations which can be represented as structure transformations. A structure operational language is a helpful tool for precisely specifying and for experimenting with heuristic picture processing strategies.

Languages to implement picture processing algorithms are divided into two categories: descriptive graphical languages and graph-structure processing languages.

Descriptive graphical languages have tended to be display-oriented and of limited use in recognition of pictures. In this class are the systems and languages of Herzog [1], Kulsrud [2], and Williams [3]. In Schwebel [4], some criteria for graphic languages are presented and a language, ICON, to meet these criteria is defined.

Graph-structure processing languages may be distinguished by the presence of operations which allow analysis of descriptions. Chase [5] uses a system for graph manipulation. Graph-structures of greater generality can be treated with languages given in Pratt and Friedman [6], Earley [7], Lieberman [8], Wolfberg [9], and Crespi-Reghiggi and Marpurgo [10]. These languages are considered precursors of our Structure Operation

Language, SOL, which we describe in Section 4.2.2.3 [11].

References

- [1] B. Herzog, "Lectures on Computer Graphics," Computer and Program Organization--Fundamentals, University of Michigan Engineering Summer Conference, June, 1967
- [2] H. E. Kulsrud, "A General Purpose Graphic Language," CACM, vol. 11, no. 4, April, 1968, pp. 247-254.
- [3] R. Williams, "A Systematic Method for the Creation of Data Structures in Computer Graphics Application," Tech. Report No. 403-19, EE Dept., New York University, April, 1971.
- [4] John C. Schwebel, "Towards the Specification of a New Image Processing," DCS File No. 788, U of I at Urbana-Champaign, February, 1969.
- [5] S. M. Chase, "Analysis of Algorithms for Finding All Sparsing Trees of a Graph," DCS Report No. 401, University of Illinois at Urbana-Champaign, 1970.
- [6] T. W. Pratt and D. P. Friedman, "A Language Extension to Graph Processing and its Formal Semantics," CACM, vol. 14, no. 7, July, 1971, pp. 460-7.
- [7] J. Early, "Towards an Understanding of Data Structures," CACM, vol. 14, 1971, p. 617.
- [8] R. N. Lieberman, "RSVP Relational Structure Vector Processor," Tech. Report No. 69-87, Computer Science Center, University of Maryland, Maryland, 1969.
- [9] M. S. Wolfberg, "An Interactive Graph Theory System," Report No. 69-25, Moore School of E.E., University of Pennsylvania, June, 1969.
- [10] S. Crespi-Reghizzi and R. Marpurgo, "A Language for Treating Graphs," CACM, vol. 13, no. 5, May, 1970, pp. 319-23.
- [11] A. E. Masumi, "SOL-Structure Operation Language," to be published.

4.2.2.1 SHOW-AND-TELL

A. System Objectives

The Show-and-Tell (S&T) programming system is a console-oriented software package providing a facile method of using some of the operational components of the Illiac III computer. The system operates on the hardware described in Section 4.2.3.1. Overall documentation of the Show-and-Tell system is contained in reference [1].

S&T is designed to support local image filtering and image acquisition directly using the Illiac III PAU and S-M-V Systems, and also to support experimentation in image processing theory using the high-level languages and mass storage of the IBM 360/75. To this end, a bidirectional link with the IBM 360 is provided allowing:

- (1) calling IBM 360 programs and subroutines from the PDP/8e console teletype, and
- (2) transmission of data and pictures back and forth between the PDP/8e and the IBM 360.

Because of the variability and unpredictability of most pictorial data, theoretically established methods and analyzed approaches frequently fail to work. Development of better theory and implementation of effective algorithms require that a programming system for experimentation in image analysis support a "trial-and-error" mode of operation. For this reason, the Show-and-Tell System design includes the following features:

- (1) Programs are converted to an interpretable linked-list structure which can be modified during algorithm development without recompiling

the entire program;

(2) Individual instructions can be tried out for effect out-of-line with the rest of the program, and then inserted if found to be helpful;

(3) Management of picture and other data is designed to be as automatic as possible, with storage allocation and data conversion handled by the system;

(4) Development of debugging aids is facilitated by the object program format, since source statements can be reconstructed for display.

A final design objective of this work is to implement a set of primitives with as general as possible applicability, so that more complex functions can be easily constructed and altered as higher-level requirements become clearer, and so that the same set of low-level building blocks can be used for a variety of practical problems.

B. System Elements

Show-and-Tell is currently composed of the following PDP/8e-resident subsystems and IBM 360-resident subsystems:

PDP/8e-resident subsystems:

- Translator: Converts S&T language typed by the operator into an interpretable list structure. Enough information is contained in this internal coding to reconstruct the source statements on demand for listing and editing.
- Interpreter: Operates on the aforementioned list structure and on the data storage lists.
- Interpretable Code Area (ICA): Storage for the list structure produced by the Translator.
- Executable Code Area (ECA): Storage for programs executed

directly by the PDP/8e hardware; i.e., the output of the PAL assembler.

- Interpretable Code Loader: Loads interpretable code into the ICA.
- EC Loader: Loads executable code into the ECA.
- Supervisor: Provides first-level interrupt handling and controls loading of System Command Processors.
- System Command Processors: Carry out system commands.
- Data Storage Lists: Contain various predefined data types in automatically maintained lists.

IBM 360-resident subsystems:

- PAXDRIVR: Accepts commands and subroutine calls from the PDP/8e, reformats argument strings for FORTRAN compatibility, executes subroutines.
- S&T/360 subroutine package: FORTRAN-callable sub-routines to permit the IBM 360 to perform exchange of data with the PDP/8e. Also included are some S&T-compatible versions of PAX II subroutines.

C. System Status and Proposed Developments

Presently two versions of the system exist:

1. a remote version which includes linkage to the IBM/360 via PAXDRIVR (see Section B).
2. a local version which includes a PAU simulator to perform the image filtering operations, but does not have the PAXDRIVR links.

The remote version has most of the operations defined in [1] debugged and has been already used as a research tool, primarily in the development

of programs for analysis of cervical smears [2] and in research in texture analysis [3]. The actual image processing is done on the IBM 360/75 of the Computing Services Office, with the S&T software performing image acquisition and display, picture transmission back and forth between the 360 and the PDP8/e, and supporting operator interaction to permit empirical establishment of various parameters. The local version is in the process of being implemented.

References

- [1] Read, John S., DCS Report No. 429, "Show-and-Tell System Specifications," University of Illinois, Urbana, Illinois, March, 1971.
- [2] Read, John S., DCS Report No. 497, "Parallel Image Processing for Automated Cytology," (MS Thesis), University of Illinois, Urbana, Illinois, 1972.
- [3] Read, John S. and Sadali N. Jayaramamurthy, "Automatic Generation of Texture Feature Detectors," to appear in IEEE Transactions on Computers, July, 1972.

4.2.2.2 PAX LANGUAGE

The PAX picture processing language (currently PAX II) represents one of the two major languages for low-level picture processing. As such, the language was discussed extensively at the IFIP Graphical Languages Conference, Vancouver, B.C., May 21-25, 1972.

It consists of 111 universal (i.e., usable at any installation running FORTRAN IV) sub-routines with 147 entry points. The core of PAX II (the original PAX was written at the University of Illinois) was written at the University of Maryland.

The Illiac III project is currently maintaining the IBM system 360 version of PAX II. A PAX Users Newsletter, sent to 45 people at 22 installations, was initiated with the collaboration of the University of Maryland.

As a result of the Vancouver conference, a formal proposal is being drafted to set up an international commission to study and advise on the design of a new graphical language (an ALGOL of image processing and computer graphics). Representatives of six countries currently are supporting the resolution. B. H. McCormick was asked to draft the proposal in recognition of his pioneering work on the PAX picture processing language.

4.2.2.3 SOL: STRUCTURE OPERATION LANGUAGE

A. Introduction

The Structure Operation Language, SOL, is a computer language designed to implement graph-structure representations and necessary operations for performing structure transformations. SOL can thus be used as a picture processing language operating on graph-structured picture representations. Numerous other applications for SOL exist in areas where a graph or graph-structured representation is used.

SOL may be described as a graph-structure processing language. It does not fall into the class of linguistic or syntactic-oriented picture processing models or descriptive graphic languages, which were included in the survey of section 4.2.2.

In this work, there is no explicit picture description language for describing pictures independently of processing algorithms. It seems realistic to first define the structure requirements which are forced by processing algorithms, and not to fix (by description languages and recognition systems) a complete picture processing model. This gives us the complete flexibility of using SOL to design a system for analyzing any specific classes of pictures.

A.1 Implementation

Currently we are implementing the SOL language by embedding it in procedural language PL/1. The major criterion of the syntax design and implementation is compatibility with PL/1. In section B we give a complete definition of the SOL and in section C we give its syntax. Any syntactical element not defined in this description will by default be assumed to be compatible with PL/1, and all its data structures are assumed to be automatic. Data structures are implemented using Linear Lists and a Table structure

using Controlled and Based variables of PL/1. The SOL preprocessor will translate the SOL statements into PL/1 source, and the output is subject to compilation by PL/1.

A.2 Language Requirements

The following requirements which are essential to any graph-structure operation language are ratified in the design and implementation of the language. These requirements will be listed in two parts: first, the elements of the structure, and secondly, the requirements of the elements necessitated by the dynamic nature of the processing.

Static Requirements

- Basic elements
- Primitives (nodes)
- Relations (branches) between pairs of nodes
- Attributes and values for nodes
- Attributes and values for branches

Sets of Elements

- Arbitrary sets of nodes and branches
- Graphs and subgraphs
- Graph-levels (explicit substructures)
- Pointers to elements
- Attributes and values for sets of elements

Dynamic Requirements

- Add-delete operations
- Functions on attribute values
- Pointer move operations
- Structure replacement operations

B. Definition of SOL

B.1 SOL Statements

The Structure Operation Language consists of statements called graph-structure statements which are embedded in a PL/1 program. SOL statements consist of declarations, associations, and operations. Declarations declare the basic structure elements. Associations declare and associate any PL/1 variable with the basic structure elements. Operations are performed on the declared structures and are capable of dynamically creating new structures.

B.2 Basic Structure Elements

There are five basic structure elements which are designated by the attributes in a declaration. The elements are the pointer, set, node, branch, subgraph, and graph.

A pointer points to or designates any other basic element. A pointer can designate a graph, branch, node, set, or pointer. Thus, a pointer allows indirect reference to any element.

A set designates a set of basic elements. It is a collection of pointers, each of which designates one element.

Nodes and branches are elements of graphs. A node designates a set of branches which are called adjacent branches of the node. A branch designates a pair of nodes. An oriented branch designates an ordered pair of nodes, the tail and the head of the branch. An unoriented branch designates an unordered pair of nodes. A subgraph is a collection of nodes in a graph.

A graph is a set of nodes and branches with the property that if a branch belongs to a graph then its tail and head also belong to the graph. Graphs will be assumed to have oriented branches unless specifically declared otherwise.

B.3 Declarations

The DECLARE statement is used to declare the five basic structure elements. An element is declared by an identifier followed by an attribute which specifies the type of element. Declaring an element creates a new data element with the name given by the identifier. A level hierarchy is assumed within a declaration such that any graph element declared is assumed to belong to the last declared element at a lower numbered level. The level numbers are assigned so that $GR < ND = BN$. Also, graphs will be assumed to be subgraphs of the first preceding graph declared in the same declaration. Thus, nodes and branches will always belong to the last declared graph.

The attribute of a branch may be of the form $BRANCH(N1,N2)$ where $N1$ and $N2$ are names for the tail and the head of the branch, respectively. This declaration will also declare the head and tail as nodes so that they need not be declared separately. The branch may also be declared without head and tail, which may be specified later, for example, by a LINK operation. Example declarations are given in Figure 1.

DECLARE

```
G1 GRAPH,  
B1 BRANCH(N1,N2),  
H SUBGRAPH,  
B2 BRANCH(N2,N3),  
B3 BRANCH(N4,N2),  
B4 BRANCH(N3,N4);
```

DCL

```
G2 GR,  
N4 ND, B2 BN,  
B1 BN(N2,N1), B1 BN(N4,N3),  
B2 BN(N1,N3), B2 BN(N2,N4);
```

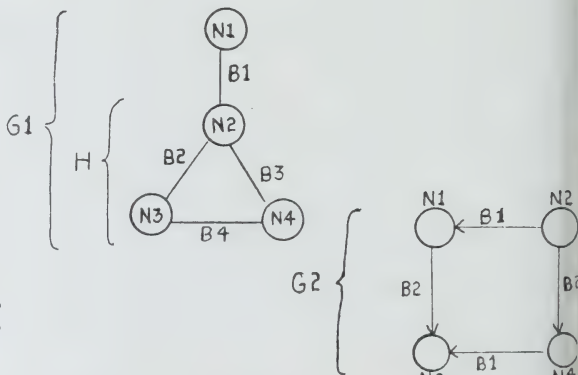


Figure 1. Example Declarations

3.4 Names and References to Elements

Names are not required for nodes and branches even though they are necessary in the DECLARE statement. Operations which dynamically modify a structure can result in the creation of a new unnamed element. Names need not be unique. For example, many branches can have the same name. Elements with non-unique names may be referenced uniquely, for instance by their position relative to other elements.

By a reference to an element, we mean a unique way of referring to the element. An element can be referenced by its name when the name is unique within the current scope or context. An element can always be referenced by a pointer, since a pointer always points to only one element. Names for graphs, pointers and sets are always references, i.e. unique within the current scope, so a pointer or a set can always be referenced by a name.

Subgraphs, nodes, and branches, however, may have non-unique names. In this case, they may be referenced by a pointer, by a qualified name, or by other contextual means. A PL/1 qualified name may be used for subgraphs, nodes, and branches using the level hierarchy assumed in the declare statement. A graph reference, for example, as used in a subgraph-operation, may explicitly list the elements of the subgraph. In this case, once the graph context is established, non-unique names may be used for references to nodes and branches.

3.5 Associations

An associate statement allows PL/1 variables to be declared and associated with any or all basic structure elements of a given type. The free statement nullifies the association. Thus, an association is a function which can be defined on any or all elements of one type and which has a value of

the type specified within the PL/1 declaration.

The list of element-names specifies the elements to be associated with or freed from the variables which follow. If the element-name list is absent, all elements of the type specified are used. For example, given the example declaration of Figure 19, the associations:

```
BNASOC RELTYP BIN FIXED;
```

```
NDASOC (N1,N2,N4) XP BIN FIXED, YP BIN FIXED;
```

would associate a variable RELTYP of type fixed binary with all the branches of the graph G2 and would associate fixed binary variables XP and YP with the nodes N1, N2, and N4. The statements BNFREE RELTYP; NDFREE XP, YP; would free the elements from the association.

The value of a variable associated with an element is accessed by using the variable (or function) name followed by an element reference enclosed in parentheses. In the preceding example, XP(N2) would refer to the value of the variable XP associated with the node N2, (N2 → XP).

B.6 Data-Operation

Data-operations dynamically add or delete elements to or from previously declared graphs, subgraphs, and sets. A reference to the element being modified appears in parentheses following the operation name. The identifier and attributes which follow refer to new elements to be added or elements to be deleted.

Deleting an element from a set does not destroy the element but only deletes the reference to the element contained in the set. Similarly, deleting a node or branch from a subgraph will not destroy the node or branch, since it will still be a member of some graph. Deleting an element from a graph which is not a subgraph of another graph, however, will delete the

reference to the graph element and remove the element from existence.

For example, `ADD(G1) N5 NODE, B5 BRANCH(N5,N2);` will add one node and one branch to the graph G1 of the previous example.

`DEL(G1) H;` will delete the subgraph H from the graph G1 and remove all the nodes and branches of H from existence.

`ADD(G2) S3 GRAPH, BA1 BRANCH(N2, NN), BA2 BRANCH(NN,N3);` will add a subgraph S3 with two new branches, BA1, BA2, and one new node, NN, to graph G2.

`ADD(S) P;` where S is a set and P is a pointer to some element, will add that element to the set S.

Deleting a node or a branch will also delete the node or branch from the sets which refer to it.

B.7 Loop-Control

The loop-control statement allows execution of the statements between the FOR statement and the END statement iteratively. `FOR (i,v,s)` specifies that each iteration will have a different value of the variable v chosen from the set s. The variable i specifies the number of iterations to be performed. ANY is equivalent to 1, and ALL is equivalent to the cardinality of the set s. Changing the set s within the loop can change the number of iterations performed. For example, if an element x is deleted from s before x is used as the value of the loop-control variable v, then x will not be used.

B.8 Other Operations

Sets-set are binary operations on sets which return a set. The set returned is the union, intersection, difference, or symmetric difference of the given sets.

Sets-Boolean return a Boolean value corresponding to the truth or falsity of the statements " s_1 equals s_2 ," " s_1 is a subset of s_2 ," or " s_1 is a subgraph of s_2 ." The functions NODES and BRANCHES operate on a graph or set and return the set of nodes or the set of branches of the graph or set.

The INCBR, OUTBR, and ADJBR functions operate on a node and return the set of incoming, outgoing or adjacent branches of the node, respectively.

The HEAD and TAIL functions operate on a branch and return the head and tail of the branch respectively.

The CARD function operates on a set and returns an integer which is the cardinality of the set.

The NAME function returns the name of the element referenced. If a pointer name is used as the reference, then the name returned is the name of the element pointed to by the pointer.

The TYPE function returns the type of the element referenced. The type is "GR," "SG," "ND," "BN," "PT," or "ST."

B.9 Pointer-Operation

Pointer-operations change the value of a pointer. They are primarily useful for moving a pointer to adjacent nodes and branches of the current element pointed to, or for moving a pointer anywhere on a graph without having to use names.

The MOVE operation moves the pointer to an adjacent branch if it points to a node, or to an adjacent node if it points to a branch. If the branch (node) name is given, the pointer can only be moved to a branch (node) with the given name. If the named branch (node) does not exist as an adjacent branch (node), the pointer is not changed.

The OMOVE or "oriented move" operation is similar to MOVE except that moves will be made along branches only in the tail-to-head direction.

The JUMP operation moves the pointer to any node or branch on the graph. JUMP will move only from a node to a node or a branch to a branch so that the type of element pointed to is not changed. If no name is present, an arbitrarily picked node (branch) is used. If a name is present and no node (branch) with that name exists in the graph, then the pointer is not changed.

The MOVE2 operation is similar to two consecutive moves, so that a node pointer is moved to an adjacent node or a branch pointer is moved to an adjacent branch. MOVE2 is not equivalent to consecutive MOVE's since the operation will be performed completely or not at all. Thus, for example, MOVE2 PTR1 RIGHT PLANE; will move a pointer named PTR1 along a branch named RIGHT to a node named PLANE only if both the branch and the node exist. However, the sequence MOVE PTR1 RIGHT; MOVE PTR1 PLANE; can result in only moving the pointer to a branch named RIGHT.

Assume a pointer named BUG points to a node. Then the operation:
MOVE BUG BRANCH2; moves BUG to an adjacent branch named BRANCH2.
MOVE BUG; moves BUG to an adjacent node.
MOVE2 BUG RIGHT; moves BUG along a branch named RIGHT to an adjacent node.

B.10 Higher-Level Graph Operations

Node-operations and subgraph-operations both deal with subgraphs within a graph. Node-operations operate on a node and can partition a node into a subgraph or generate a subgraph sub-structure from a node. Subgraph-operations deal with a subgraph specified by a node set. The subgraph can be merged into one node, linked by "SUBPART" branches to one node, disconnected from all outside nodes, or be linked by a node chain. Figure 2 illustrates some of these operations.

The MERGE operation operates on all nodes of a specified subgraph and results in a new node. The new node may be named or may be pointed to by assigning the result of the MERGE to a node identifier or to a pointer, respectively. The following actions occur when nodes XI are merged into a node X:

1. A new node, X, is created.
2. The node associations for X are the union of the associations for the set XI, and the value of each association is the same as the value for some arbitrary element of XI.
3. All branches between any element in XI and any element not on XI are changed to link between the same element not in XI and the new node X.
4. All nodes of the subgraph and all branches between nodes of the subgraph are deleted.

The PARSE operation is similar to the MERGE operation except that the merged subgraph is not deleted and the merged nodes are linked to the new node by "SUBPART" branches.

The PARTITION operation specifies a node and a partitioning of that node into nodes and branches. It is the inverse of the MERGE operation. The node specified is replaced by the subgraph and a procedure is necessary to specify the embedding (branches) between the subgraph and the graph.

The GENERATE operation is similar to the PARTITION operation except that the partitioned node is retained. The nodes of the subgraph specified are linked to the node specified by "SUBPART" branches.

The DISCONNECT operation disconnects all branches between any node outside and any node inside the set of nodes specified. If a branch name is specified, only branches with the given name are deleted. SINK and SOURCE

operate like DISCONNECT except that SINK does not disconnect branches directed into the node set, and SOURCE does not disconnect branches directed out of the node set.

The LINK operation links all nodes into an arbitrarily ordered chain by branches having the specified name. If no branch name is specified, unnamed branches are used.

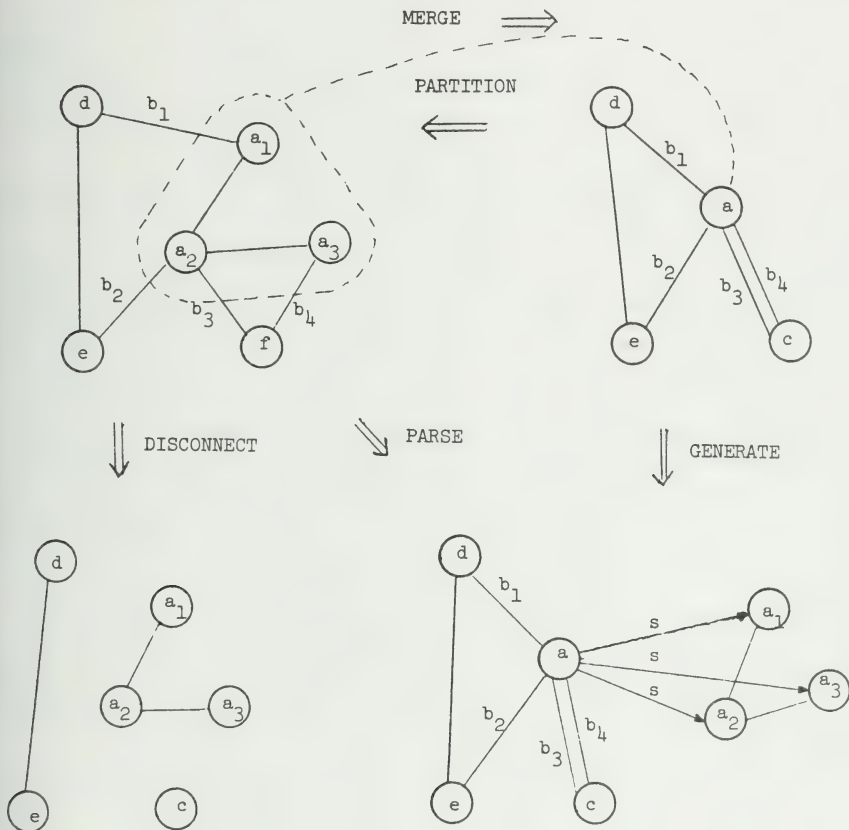


Figure 2 SOL Subgraph Operations

The TRANSFORM operation will replace a subgraph (domain of transformation) of the data graph, by another graph or subgraph. Embedding of relations between the nodes of new subgraphs and the nodes external to the domain are specified by a procedure call. We do not keep the memory of newly formed nodes by subpart links.

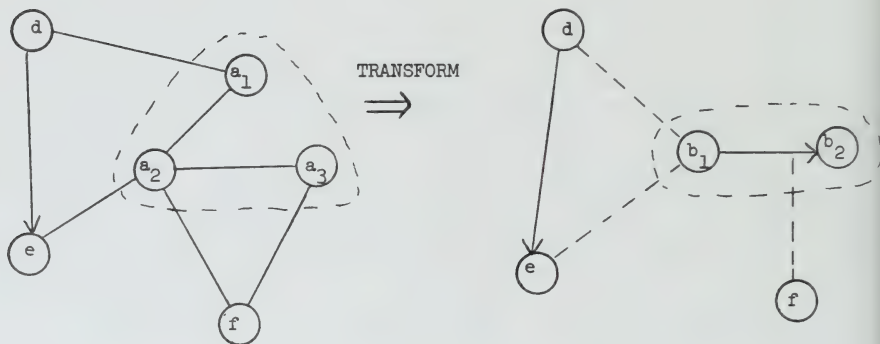


Figure 3 Transform Operation

C. SOL Syntax

The syntax of SOL is given below in PL/1 syntax notation. Any non-terminal beginning with "pli" is the same as the corresponding PL/1 construct and is not defined here.

`<Sol-program> ::= <labels> {PROCEDURE | PROC}`

`<statement> [; <statement>]...`

`<statement> ::= <Sol-statement> | % <Pl1-statement>`

`<Sol-statement> ::= <labels> {<block-sta> | <declaration> | <associate> |
 <free> | <operation> | <if-statement> | <assignment> |
 <go-to-sta> | <end-sta> | <call-sta> };`

`<block-sta> ::= {BEGIN | {PROCEDURE|PROC} [(<*I> [, <*I>]...)] | DO}`

`<end-sta> ::= <labels> END <labels>`

`<go to-sta> ::= {GO TO | GOTO} <*I>`

`<call-sta> ::= CALL <*I> [(<*I> [, <*I>]...)]`

`<*I> ::= identifier`

`<declaration> ::= {DECLARE | DCL} <element-declaration> [, <element-declaration>]...`

`<element-declaration> ::= <*I> <element-attribute>`

`<element-attribute> ::= {GRAPH | GR} [UNORIENTED] | {ND | NODE}
 {BN | BRANCH} [(node-ref, node-ref)] |
 {SG | SUBGRAPH} [DISJOINT] | {PT | POINTER} | {ST | SET}`

`<if-statement> ::= IF pli-expression THEN <statement> [ELSE <statement>]`

`<associate> ::= {GRASOC | SGASOC | NDASOC | BNASOC | PTASOC | STASOC}
 [(<qualified-name> [, <qualified-name>]...)] <declaration-tail>`

`<declaration-tail> ::= <element-dec> [, <element-dec>]...`

`<element-dec> ::= [<*N>] <*I> [(<*S>)] [, <*N> <*S>]...`

`<free> ::= {GRFREE | SGFREE | NDFREE | BNFREE | PTFREE | STFREE} [(<qualified-
 name> [, <qualified-name>]...)] <*I> [, <*I>]...`

$\langle \text{operation} \rangle ::= \langle \text{loop-control} \rangle \mid \langle \text{data-operation} \rangle \mid \langle \text{sets-set} \rangle \mid \langle \text{sets-boolean} \rangle \mid$
 $\langle \text{graph-set} \rangle \mid \langle \text{node-set} \rangle \mid \langle \text{branch-node} \rangle \mid \langle \text{set-integer} \rangle \mid$
 $\langle \text{element-string} \rangle \mid \langle \text{pointer-operation} \rangle \mid \langle \text{node-operation} \rangle \mid$
 $\langle \text{subgraph-operation} \rangle$

$\langle \text{data-operation} \rangle ::= \{ \text{ADD} \mid \text{DEL} \} \{ \text{graph-ref} \mid \text{subgraph-ref} \mid \text{set-ref} \}$
 $\langle \text{qualified-name} \rangle [\langle \text{element-attribute} \rangle] [, \langle \text{qualified-name} \rangle [\langle \text{element-attribute} \rangle]] \dots$

$\langle \text{loop-control} \rangle ::= \text{FOR} \{ \text{ANY} \mid \text{ALL} \mid \langle *N \rangle \}, \langle *I \rangle, \langle \text{set-ref} \rangle$

$\langle \text{sets-set} \rangle ::= \langle \text{sets-set-mnemonic} \rangle (\text{set-ref}, \text{set-ref})$

$\langle \text{sets-set-mnemonic} \rangle ::= \text{UNION} \mid \text{INTER} \mid \text{DIFF} \mid \text{STMDIFF}$

$\langle \text{sets-boolean} \rangle ::= \langle \text{sets-boolean-mnemonic} \rangle (\langle \text{set-ref} \rangle, \langle \text{set-ref} \rangle)$

$\langle \text{sets-boolean-mnemonic} \rangle ::= \text{EQUALS} \mid \text{SUBSET} \mid \text{SUBGRAPH}$

$\langle \text{graph-set} \rangle ::= \langle \text{graph-set-mnemonic} \rangle (\langle \text{graph-ref} \rangle, \langle \text{graph-ref} \rangle)$

$\langle \text{graph-set-mnemonic} \rangle ::= \text{NODES} \mid \text{BRANCHES}$

$\langle \text{node-set} \rangle ::= \langle \text{node-set-mnemonic} \rangle (\langle \text{node-ref} \rangle)$

$\langle \text{node-set-mnemonic} \rangle ::= \text{INCBR} \mid \text{OUTBR} \mid \text{ADJBR}$

$\langle \text{branch-node} \rangle ::= \langle \text{branch-node-mnemonic} \rangle (\langle \text{branch-ref} \rangle)$

$\langle \text{branch-node-mnemonic} \rangle ::= \text{HEAD} \mid \text{TAIL}$

$\langle \text{set-integer} \rangle ::= \langle \text{set-integer-mnemonic} \rangle (\langle \text{set-ref} \rangle)$

$\langle \text{set-integer-mnemonic} \rangle ::= \text{CARD}$

$\langle \text{element-string} \rangle ::= \langle \text{element-string-mnemonic} \rangle (\langle \text{element-ref} \rangle)$

$\langle \text{element-string-mnemonic} \rangle ::= \text{NAME} \mid \text{TYPE}$

$\langle \text{element-ref} \rangle ::= \langle \text{graph-ref} \rangle \mid \langle \text{set-ref} \rangle \mid \langle \text{node-ref} \rangle \mid \langle \text{branch-ref} \rangle \mid$
 $\langle \text{pointer-ref} \rangle$

$\langle \text{graph-ref} \rangle ::= \langle \text{qualified-name} \rangle \mid \langle \text{pointer-ref} \rangle$

$\langle \text{node-ref} \rangle ::= \langle \text{branch-node} \rangle \mid \langle \text{qualified-name} \rangle \mid \langle \text{pointer-ref} \rangle$

$\langle \text{branch-ref} \rangle ::= \langle \text{qualified-name} \rangle \mid \langle \text{pointer-ref} \rangle$

$\langle \text{set-ref} \rangle ::= \langle \text{sets-set} \rangle \mid \langle \text{graph-set} \rangle \mid \langle \text{node-set} \rangle \mid \langle *I \rangle \mid \langle \text{pointer-ref} \rangle$
 $\langle \text{pointer-ref} \rangle ::= \langle *I \rangle$
 $\langle \text{qualified-name} \rangle ::= \langle *I \rangle [, \langle *I \rangle] \dots$
 $\langle \text{pointer-operation} \rangle ::= \{ \text{JUMP} \mid \text{MOVE} \mid \text{OMOVE} \} \langle \text{pointer-ref} \rangle [\langle \text{branch-ref} \rangle \mid$
 $\quad \langle \text{node-ref} \rangle] \mid \{ \text{MOVE } 2 \mid \text{OMOVE } 2 \} \langle \text{pointer-ref} \rangle$
 $\quad [\langle \text{branch-ref} \rangle [\langle \text{node-ref} \rangle] \mid \langle \text{node-ref} \rangle [\langle \text{branch-ref} \rangle]]$
 $\langle \text{node-operation} \rangle ::= \{ \text{PARTITION} \mid \text{GENERATE} \} \langle \text{node-ref} \rangle \langle \text{graph-reference} \rangle$
 $\quad \langle \text{call-sta} \rangle$
 $\langle \text{subgraph-operation} \rangle ::= \{ \text{MERGE} \mid \text{PARSE} \mid \text{DISCONNECT} \mid \text{SINK} \mid \text{SOURCE} \mid \text{LINK} \}$
 $\quad \{ \langle \text{node-ref} \rangle \mid \langle \text{set-ref} \rangle \} \dots [\langle \text{branch-ref} \rangle] \mid \text{TRANSFORM}$
 $\quad \{ \langle \text{graph-ref} \rangle \mid \langle \text{set-ref} \rangle \} \{ \langle \text{graph-ref} \rangle \mid \langle \text{set-ref} \rangle \}$
 $\quad \text{EMBED } \langle *I \rangle$
 $\langle *N \rangle ::= \text{integer}$
 $\langle *S \rangle ::= \text{character string}$
 $\langle \text{ASSIGNMENT} \rangle ::= [\#] \langle *I \rangle = \langle \text{qualified-name} \rangle$

4.2.3 SCAN/DISPLAY

4.2.3.1 DESCRIPTION OF OPERABLE EQUIPMENT

The Show-and-Tell system (c.f. 4.2.2.1) is currently using all the operable scan/display system equipment. This equipment is shown schematically in Figure 4. The various subsystems illustrated are discussed below.

- * DEC PDP8/e Subsystem: This subsystem consists of a Digital Equipment PDP8/e computer with 8K of core (12 bits/word) and two teletypes (one is remotely located). There are also three interfaces designed and constructed during this past year; the interface to the IBM 360/75 (c.f. section 4.2.3.2), a 2.4 megabit interface to the Illiac III Exchange Net, and a Linctape controller capable of driving up to eight transports (currently supporting two transports).

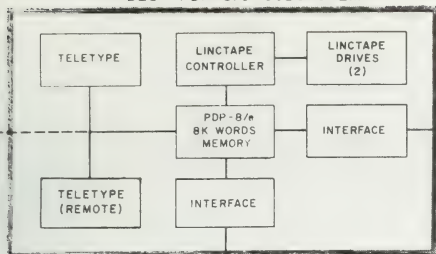
- * Illiac III Core Machine: This section consists of the Exchange Net and 256 K Bytes of fast core storage. It also contains a very simple I/O Processor.

- * SMV Subsystem: This section consists of one SMV controller with two scanning stations and a display monitor. The scanning stations consist of a 46 mm flying spot film scanner (now being converted to 35 mm) and an automated stage flying spot microscope scanner.

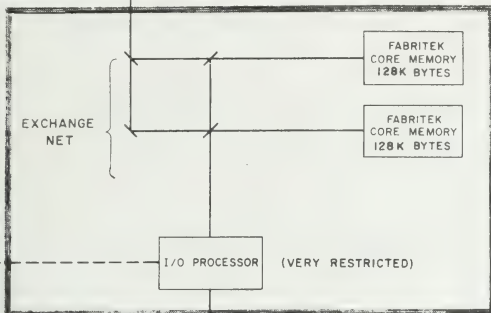
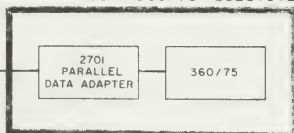
- * Video Subsystem: This section comprises a manual video switching network and three high resolution television systems (1536 lines); an automated microscope, a manual microscope and a large format system (suitable for text, x-rays, etc.). The video subsystem is currently being attached to the SMV subsystem.

- * IBM 360/75 Subsystem: This section consists of the University's Computer Service IBM 360/75 running the PAX II image processing system.

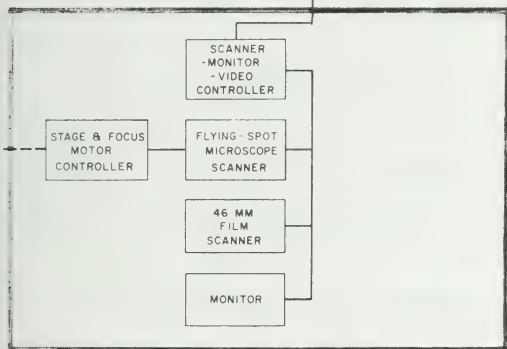
DEC PDP-8/e SUBSYSTEM



IBM 360/75 SUBSYSTEM

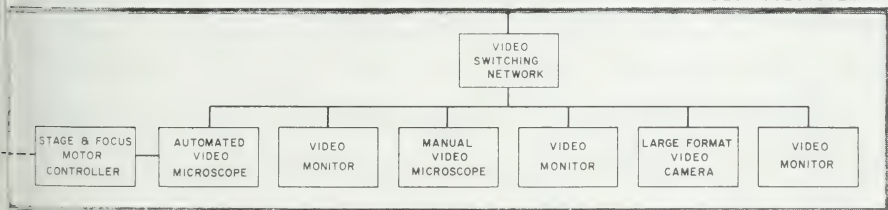


ILLIAC III CORE MACHINE



S-M-V SUBSYSTEM

VIDEO SUBSYSTEM



— DATA AND/OR CONTROL

- - - CONTROL ONLY

FIGURE 4- OPERABLE EQUIPMENT

4.2.3.2 PDP8/e CONTROLLER AND INTERFACE TO THE IBM 360/75

In order to facilitate image processing research, it was decided to add a link to the University's Computer Service IBM 360/75 via a 2701 Parallel Data Adapter. This link is used to transmit pictures to the PAX II system on the 360/75, where they are processed interactively, and then re-transmit the processed images back for display.

The interface was designed and constructed during the last year and has been operating without error since December.

4.2.3.3 AUTOMATED MICROSCOPE

The flying spot microscope became operational during this past year. Full control of X, Y, and Z axis motion is possible and under PDP-8e program control. X and Y steps move the slide by $34.4\mu\text{m}$ increments, while the Z-axis drive (focus) moves the slide by $0.12\mu\text{m}$ per step. The stage motors may be driven at up to 200 pps bidirectionally.

A five lens turret will take any standard microscope objective (three powers are currently provided). A scheme was devised to add a reference photomultiplier before the spot image passed through the objective by using one of the optical charger parts. This was expedited, and an improved image was obtained since product detection was then possible. Resolution is very good but as yet unmeasured.

The CRT currently in use in this device is a Litton L-4238 with P-16 phosphor. This tube is being operated at an excessively high beam current to overcome the light loss in the microscope optical system due to a small entrance pupil and to absorption of the CRT's blue UV light output by the

large number of glass elements. It is anticipated that this situation can be improved by replacing the P-16 equipped CRT with a tube using one of the new broad-spectrum phosports; this will have the additional benefit of enabling experimentation with color data.

A complete description of the microscope stage control may be found in File No. UIUDCS-F-72-871, "IMAGE 8: Microscope Stage and Focus Motor Control," by John Read.

4.2.3.4 VIDEO DIGITIZATION

The fabrication of the second SMV control, necessary to digitize video, was completed. Checkout will begin using the current monitor. When the control is checked out, it will be modified to handle the higher video data rates.

The actual digitizer circuitry has been designed and laid out. It will accept the video, with sync signals, as input and output analog signals which are equivalent to the normal PMT output, as well as digital control signals for use by the SMV control.

4.2.3.5 46 mm SCANNER CONVERSION

In order to alleviate the problem of reformatting 35 mm film into the 46 mm format for the operational 46 mm film scanner, it was decided to convert the second 46 mm film transport to handle 35 mm sprocketed film directly.

The unit will handle standard 35 mm sprocketed film in the full frame (ie double frame) format found in most still cameras. Any length from a 3 exposure strip to a 400 foot roll can be loaded and examined (strips must be

spliced in, or "clipped" in if they have sufficient non-image extra length).

All transport functions will be under PDP-8e control. The film may be bi-directionally advanced by as little as .08 sprocket holes or .015 inches at a maximum of 200 steps per second. Using normal 8 sprocket per frame film, this leads to about one half second per frame advance, which is considered adequate. Film tensioning, clamp, and end-of-film flagging are automatic. Left and right end-of-film infra-red sensing will set a flag and disallow advance in the wrong direction. As there is a rather simplified control scheme for the film tension motors, no high speed slew or rewind is available; thus, maximum film speed will be limited to about 2 frames per second.

4.3. PATTERN RECOGNITION

Three developments here this past year in pattern recognition have excited national response:

- (1) The concept of Covering Theory, with its extension of the methodology of switching theory to the characterization of n-dimensional discrete sets. The utility of this concept, and the operable analysis algorithms, would appear to be a major innovation. Results related to covering theory are reported in Section 3.1 (available reports are listed there). Three operable PL/1 programs for the IBM 360/75 are available:

AQ-3i for the synthesis of interval covers [1].

AQ-4c for the synthesis of cartesian, interval or "mixed" covers (the latter are covers where some literals are cartesian and some are interval) [2].

AQ-5cm for the synthesis of cartesian covers of a disjoint family of event sets (only a very simplified version of the program is presently in operation).

- (2) The extension of covering theory provides a foundation for Vari-Valued Logic. This technique allows one to speak quantitatively of a "simple" description--where the computational cost of all available descriptions in the language can be assessed. Rather than a statistical approach, emphasis here is upon descriptive minimization--to identify which features in which logical construction make discrimination of two (or more) sets possible. Recently it has also become possible to speak formally of "analogy" and mechanize the search for analogy.

- (3) Continued expansion of the Signal Detection Theory to discrete sets of local patterns--most often to attain texture discrimination/analysis--has continued unabated this past year (c.f. 4.4.3).

References

- [1] R. S. Michalski, Val Tareski, "Interval Covers Synthesis--Computer Implementation of and Experiments with Algorithm A^Q," to be published.
- [2] R. S. Michalski, P. Raulefs, "Computer Synthesis of Cartesian Covers," to be published.

4.3.1 COVERING THEORY

In [1, 2] we introduced a concept of an interval cover of a set against another set in a discrete multidimensional space. The interval cover is a set of multi-dimensional intervals (called interval complexes), whose set-theoretical union includes every element of the first set and does not include any element of the second set. The above concept has found a number of applications, in particular in pattern recognition and picture processing [3]. The present paper [4] generalizes the above concept to that of a cartesian cover. Here elements of a cover, called cartesian complexes, are cartesian products of any subsets of the domains of given variables. Thus the interval covers are special cases of cartesian covers where the subsets of the domains of the given variables are restricted to be chains. In the paper we also introduce and discuss another special case of cartesian covers, namely cyclic covers (interval covers with complements).

Examples of practical applications of the concept of a cartesian cover are, e.g., a compact representation of n-ary relations, cluster extraction, determination of a set of complete subgraphs of a given graph, map coloring, etc. This concept is also a basic tool for the synthesis of variable-valued logic formulas [3].

References

- [1] R. S. Michalski, B. H. McCormick, "Interval Generalization of Switching Theory," Proceedings of the Third Annual Houston Conference on Computer and System Science, Houston (Texas), April 26-27, 1971 (an extended version in Report No. 442, Department of Computer Science, University of Illinois, Urbana, Illinois, May 3, 1971).
- [2] R. S. Michalski, "A Geometrical Model for the Synthesis of Interval Covers," Report No. 461, Department of Computer Science, University of Illinois, Urbana, Illinois, June 24, 1971.

- [3] R. S. Michalski, "A Variable-valued Logic System as Applied to Picture Description and Recognition," Proceedings of the IFIP Working Conference on Graphic Languages, May 22-26, 1972, Vancouver, Canada.
- [4] B. H. McCormick and R. S. Michalski, "Cartesian Covers: A Class of Set Representations in a Discrete Finite Space," Department of Computer Science Report (in progress), June, 1972. To be submitted for publication.

4.3.2 VARIABLE-VALUED LOGIC

A. Abstract

The paper by Michalski [1] introduces a concept of a variable-valued logic system, defines a particular system VL_1 and demonstrates, by examples, its application as a language for describing complex graphical objects and also as a tool for making inferences about significant properties of the objects or classes of objects. (Included examples show how to use the system to describe a picture treated as a structure of inter-related components, to deduce a simple rule characterizing one class of patterns as opposed to another and to synthesize a minimal set of filters for the discrimination of a collection of textures.) A brief summary of the computer implementations of the developed concepts is also included.

B. Motivation and Description of Present Work

In the conventional approach to designing graphical languages, the problems of obtaining the simplest of possible equivalent descriptions of an object (according to certain defined criteria) have not received much consideration. Yet, in the application of the descriptions to the recognition of objects, the simplicity of descriptions plays an important role. Another problem related to the recognition of objects, in particular, of pictures, is to provide a formal tool for selecting the most "significant" features of the described objects.

A possibility of having a formal system with mechanisms for obtaining the simplest (in a clearly defined sense) descriptions of objects, as well as for finding their most characteristic features, has a practical importance. The purpose of the paper is to investigate this possibility.

We present here a formal system with a few simple operations (or a language in a broader sense) which--when supplied with primitives and their relations (found in a described object or class of objects)--produces a description which is minimal in a well-defined sense. The system can also be used to deduce the simplest rule for recognizing one class in the context of another class (or classes) of objects. This system belongs to the domain of logic rather than to that of formal languages and is called variable-valued logic. Actually, in the paper we describe one particular system of the variable-valued logic, namely VL_1 . It is an outgrowth of work with Professor Bruce McCormick on the so-called interval and cartesian covers. A number of concepts included in the system are directly related to the concepts in our common reports (Michalski and McCormick (1971c, 1972b)).

Since the description of VL_1 is not available elsewhere, it was necessary to include it here, and therefore Chapter 2 is devoted to the formal definition of the system. Chapter 3 describes briefly the problem of synthesizing formulas in the system (which satisfy certain cost functionals) and summarizes the computer implementations of the developed concepts. Chapter 4 gives a number of simple examples which show, in particular, the application of the system to inferring, based on logical principles, a simple rule which characterizes one class of objects as compared to another or to finding a minimal set of filters for discrimination of textures. The last example is used to demonstrate how to apply VL_1 to describing a graphical object, viewed as a structure of interrelated components.

Reference

- [1] R. S. Michalski, "A Variable-Valued Logic System As Applied to Picture Description and Recognition," Proceedings of the IFIP Conference on Graphic Languages, May 22-26, 1972, Vancouver, Canada.

4.4. IMAGE PROCESSING THEORY

A. Descriptive Pattern Analysis

Descriptive pattern analysis attempts to build descriptions of patterns.

We refer to the procedures which form the core of this analysis as parsing procedures. The central problems attacked by these procedures are the location, isolation, and identification of objects in a picture. These problems have been referred to as the "pattern recognition problem"; however, we believe that they can be better delineated as the problems of segmenting, clustering, and parsing objects in a picture. The current lack of computational sophistication in attacking these problems is attested to by the seemingly elementary (by human standards) image processing that systems today are able to perform.

In our efforts to develop parsing procedures, we have tried to find techniques applicable to the largest class of pictures first, and specialize to other procedures only when forced to do so. As a result we divide clustering procedures into two categories: first, the lower-level segmentation and clustering procedures for object isolation, and second, higher-level parsing procedures utilizing class-specific rules to identify objects within the global structural model.

The first category includes classical uni-relational graph-theoretic clustering techniques as well as the newer methods of multi-relational clustering, subgraph replacement, and graph transformational rules.

The second category involves heuristic search and global inference techniques applied to the graph structure. These seek to identify where to dynamically modify the structure and to locate optimum sites for object

identification. The goal here is to select first for identification those composite structures most likely to be valid objects.

B. Graph Transformations

The abstract labeled and directed graph, whose nodes represent objects and whose edges represent arbitrary binary relations between objects allows recognition procedures based on graph transformations. The recognition process can then be viewed as replacement rules operating on graphs.

An important class of procedures forms composites from objects in a "name-independent" manner, i.e., without having to match each element to an explicitly named prototype. These procedures can choose appropriate graph transformations given relevant properties of the relations between items. This class of procedures may be used in the lower levels of analysis in order to increase the efficiency over a model-driven analyzer.

C. Productions

Set-replacement rules used to build up structure are generalizations of grammatical productions. Predicates may be used to specify when a rule may be applied. This process is called "composite formation." A replacement rule forms a composite element from subelements of the graph. The new graph formed by new composite elements and new relations involving these elements is then the candidate for further composite formation. For successful recognition, this process should tend toward a "recognizable graph," i.e., a graph which has somehow been specified as an acceptable goal.

The general structure transformation schema can now be specified by an ordered production system whose entry points correspond to root graph operations. For example, the root operation of creating a branch, or merging two elements which may be dictated by the current picture segmentation strategy,

specifies an entry point into the production schema and thus determines possible graph transformations. These graph transformations can then specify actions calling for other root operations.

D. Processing Languages

The development of processing languages is considered a necessary part of this problem. Currently, adequate languages exist to express algorithms which operate on the array representation of pictures as binary valued elements with neighborhood connectivity relationships. The next and most natural abstraction from the array representation is a graph structure, by means of which many scene segmentation algorithms can be simply expressed. We seek to develop theory for the level of picture processing operations which can be represented as structure transformations. A structure operational language is a helpful tool for precisely specifying and for experimenting with heuristic scene segmentation strategies.

In summary, the problem dealt with here is to develop representations and parsing procedures for global level picture processing. The model to be developed should allow the inference of structural models from examples and counter-examples and permit relational inference to choose candidates for composite formation, and to embed or extend relations to newly formed elements. Successful strategies for structural inference using multi-graph scene representation are undoubtedly techniques which will have utility beyond their immediate application to scene analysis.

4.4.1 RESUME OF IMAGE PROCESSING METHODOLOGY

Our long range goal can be viewed as this: to design a system which can infer the syntactic and semantic structure of its visual environment from selected instances of objects and scenes from the environment. Implicit here is the prior existence of parallel processes for scene segmentation.

Three subareas of the global recognition problem are identified: Picture Representation, Descriptive Pattern Analysis, and Processing Languages. The primary concern here will be with global aspects of processing, but we will try to use all local information and features present in the picture, [1], [2], [3], and suggest ways of incorporating these into our global processor. Therefore, some proper preprocessing will be assumed on the picture before it is treated by the more global aspects of the model. These processes typically consist of operations which can abstract low-level primitive objects as well as find relations between objects of the scene.

Fundamental to the development of higher level picture processing procedures is the creation of a suitable picture representation for algorithm and data. This representation should express the hierarchical structures of elements with attributes and relations among them. Basic to this model are a graph-structured data representation and graph-transformational parsing procedures. The graph-structural representation model has the following features which facilitate flexibility:

Subgraphs may be expressions of topological relations which permit a deformable model.

Attribute values associated with each object may be used to tie down the model to concrete instances.

Interaction or propagation of information between parsing levels, which is needed to identify objects in context, is readily expressible.

Descriptive pattern analysis attempts to build description of patterns. We refer to the procedures which form the core of this analysis as parsing procedures. The central problems attacked by these procedures are the location, isolation, and identification of objects in a picture. The current lack of computational sophistication in attacking these problems is attested to be the elementary (by human standards) image processing that systems today are able to perform.

A labeled directed graph, whose nodes represent objects and whose edges represent arbitrary binary relations between objects, allow recognition procedures based on graph transformations. The recognition process can then be viewed as the application of replacement rules to graphs. The general structure transformation reference can now be specified by an ordered production system whose entry points correspond to root graph operations. For example, the root operation of creating a branch, or merging two elements which may be dictated by the current picture segmentation strategy, specifies an entry point into the production scheme and thus determines possible graph transformations. These graph transformations can then specify actions calling for other root operation.

References

- [1] R. S. Michalski, "A Variable Valued Logic System as Applied to Picture Description and Recognition," Proceedings of the IFIP, May 22-26, 1972.
- [2] K. Maruyama, "Shape Detection," DCS thesis in progress.
- [3] S. N. Jayaramamurthy, "Texture Analysis," DCS thesis in progress.

4.4.2 SCENE SEGMENTATION

Scene analysis consists of two major parts: picture preprocessing and structural analysis.

Scenes are viewed as being pictorial representations of an assembly of more or less well known objects. In image processing research, a scene is usually given as a digitized picture constituting the output of a scanning device. The raster points of a digitized picture are referred to as picture points. The preprocessing phase consists of partitioning the picture into sets of picture points. Subsequently, these sets have to be interpreted as representations of certain objects (structural analysis).

The basic idea of the thesis of Raulefs* is that the representation of objects in a scene is given by sets of picture points such that the local properties of points within such a set are more similar to each other than those belonging to different sets. Such a set of picture points will be called a region. The idea of partitioning scenes into regions has proven to yield a convenient data structure for subsequent processing techniques (see [1, 2, 3, and 4].

Although this approach to region definition is closely related to the concept of clustering, a precise definition has not yet been given. It is shown that the information-theoretical quantity entropy is useful in characterizing a region. Entropy is used to define the cohesion of a region, and this term gives a tool to analyze clustering methods and to specify an algorithm giving an optimal scene segmentation.

After having partitioned a picture into regions, each region must

*Peter Raulefs, "Methodological Aspects of Scene Segmentation," DCS Report No. 496, University of Illinois at Champaign-Urbana, June, 1972.

be interpreted as a part of a known object. This requires a model of how objects may be represented and which relations between these representations must hold. Then, recognition can be viewed as finding an optimal match of regions to a collection of model objects. Although we do not pursue a solution to this problem in this paper, regions are used to define a data structure which is applied to approaches to scene analysis.

References

- [1] Brice, C. R. and Fennema, C. L., "Scene Analysis Using Regions," Artificial Intelligence, vol. 1 (1970), pp. 205-226.
- [2] Guzman, A., "Analysis of Curved Line Drawings Using Context and Global Information," in Machine Intelligence, Vol. 6, Meltzer, B. and Michie, D. (eds.) Edinburgh University, 1970, pp. 325-375.
- [3] Barrow, H. G. and Popplestone, R. J., "Relational Descriptions in Picture Processing," in Machine Intelligence, Vol. 6, Melzer B. and Michie, D. (eds.), Edinburgh University Press, 1970, pp. 377-396.
- [4] Preparata, F. P. and Ray, S. R., "An Approach to Artificial Nonsymbolic Cognition," Coordinated Science Laboratory Report No. R-478, University of Illinois, Urbana, Illinois, 1970.

4.4.3 AUTOMATED ANALYSIS OF TEXTURES

Visual texture is well known to play a central role in visual perception. In the context of automated image processing, there are three problems which have received extensive attention recently, viz., texture discrimination, texture analysis, and texture synthesis. In the literature we find many methods which deal with the above problems. Most of these methods, however, have little or nothing in common and lack generality.

It is our intention to propose and develop a method which would be sufficiently versatile to deal with most of the problems in texture: viz., texture discrimination/recognition, texture analysis, and texture synthesis.

Our proposed method is based on the principles of signal detection theory. By extending this method and with the application of Interval Covering Theory recently developed at the University of Illinois by Professors B. H. McCormick and R. S. Michalski, we show how we generate texture feature detectors.

A. Brief Introduction of the Method

To serve as an introduction to the method of texture recognition, we will briefly be describing how we presently deal with this problem and later on show how it can be extended to cope with more complex problems.

We often encounter a situation where we are presented with a pair of families of visual scenes (digitized images, of course) which differ in texture, say, those of malignant and nonmalignant tumor cells. The problem is to distinguish and appropriately label any member of these families. This is the problem of texture recognition.

For this purpose, we define a template to extract patterns from the scene of analysis, i.e. n-tuple of gray levels centered around each given positioning of the template. The occurrence of any pattern can be considered as an "event." All possible patterns that can be extracted by the template defines the universe of events. With the help of samples from both texture families, we calculate statistics of each event such as, the probability of its occurrence in any given family, the "likelihood ratio" which is the ratio of the probability of its occurrence in one family to the other, and so on. Using criteria which we shall discuss in detail later in this report, we extract two disjoint sets of patterns (events). The patterns in the same set have the property that their occurrence in one family is more likely than in the other. The classification of an "unseen" sample (i.e. not included in the initial training sample) is achieved on the statistics of the occurrences of patterns belonging to the above-mentioned sets in the scene of analysis. The thresholds needed for the classification are determined with the help of the training samples. We have essentially constructed a filter by defining one set as the passband of events and the other as the stopband. A scene of analysis is classified by the statistics of the events that would fall in these bands.

A.1 Interval Complexes

Starting with the aforementioned disjoint sets and with the applications of the methods in "interval covering theory," a set of "interval complexes" are then developed. Reserving the explanation of terminology and other details for a later chapter, we shall add here only that these "interval complexes" serve as 2-D filters for textural features which are more likely to appear in one family than in the other.

A.2 Controlling Parameters

It can be seen that the analysis of texture is performed by

using patterns, which are local in nature and can only be as global as the size of the template permits. There are also other parameters other than the size and shape of the template, such as resolution, the choice of which has a significant effect on the success of the method. Developing an algorithm for an optimal choice of all these parameters, given the scenes of analysis, is one of the top priority items under current study.

A.3 Distinguishability Index

A visual aid (graph) and an index derived from it has been developed to determine how well the two families of textures can be distinguished for a certain choice of parameters. The graph is known as the R.O.C. (Receiver Operating Characteristic) Curve, as used in Signal Detection Theory. The index can vary from 0.5 for the worst case (two textures are not distinguishable for the present choice of parameters) to 1.0 for an ideal case. This index is a very helpful guide in the iterative choice of a better set of parameters.

A.4 Implementation of the Method

Most of the operations in the method suggested can be performed in parallel and it can be very easily implementable on a computer like Illiac III utilizing its parallel processing capabilities. In fact, programs have been written implementing this method in Fortran using "PAX" Subroutines which simulate the Pattern Articulation Unit of Illiac III.

B. Generation of Textural Feature Detectors

We have described a local categorizer in the previous section and in principle it could be implemented by just looking up input events in a table of events and likelihood ratios. However, for real textures and useful neighborhood sizes, the number of events in the acceptance set R could be too large to make the process practical. Also, no categorization would be performed for events not in the training set. By applying some concepts from switching theory, equivalent but much more efficient categorizers can be generated. This is accomplished by a technique analogous to switching-theoretic procedures for minimization of the disjunctive normal form of a switching function.

In the case of binary signal detection, the events from $F^{1\beta}$ can be considered a "true" set and those from $F^{0\beta}$ are considered a "false" set. The disjunctive normal form can be expressed as $\bigvee_i \xi_i(e_i)$ where ξ_i is a predicate that has output "true" when the input is a particular event, e_i , from $F^{1\beta}$, and output "false" if it is from $F^{0\beta}$. The symbol \bigvee above represents the logical "OR" of the predicates. Events from F^* are considered as don't-care events. McCormick and Michalski have developed "interval covering theory" as a generalization of switching theory [1] which permits the transplantation of much of the minimization machinery already in existence. In particular, Michalski's A^Q algorithm [2] for generation of quasi-minimal covers can be used.

B.1 Notation

To explain the method, it is necessary to introduce a few items of notation from [2]:

E is the event space as before. That is, the set of all events.

$e = (x_1, x_2, \dots, x_n)$ where $0 \leq x_i \leq h-1$.

A literal, $a_i x_i b_i$, is the set of all events $e \in E$ whose i -th component lies between a_i and b_i :

$$a_i x_i b_i = \{(x_1, x_2, \dots, x_n) \mid a_i \leq x_i \leq b_i\}.$$

An interval is a set-theoretic product of literals,

$$L = \bigcap_{i \in I} a_i x_i b_i, \quad I \subseteq \{1, 2, \dots, n\}.$$

The interval represents a "box" in hyperspace which includes all events between (a_1, a_2, \dots, a_n) and (b_1, b_2, \dots, b_n) . Note that components not specified by the interval are free to take on any integer value in $[0, h-1]$.

An interval cover of the set $F^{1\beta}$ against $F^{0\beta}$ is defined as a union of intervals, L_j , such that:

$$F^{1\beta} \subseteq \bigcup L_j \subseteq F^{1\beta} \cup F^*.$$

Thus an interval cover contains all the events in $F^{1\beta}$ plus some in F^* , but none in $F^{0\beta}$. However, the interval cover will represent this partitioning of the space of possible events much more concisely than just enumerating all the events in the acceptance set for T^1 . Also, the interval cover can classify events which were not in the training set, because of the inclusion of F^* events in the "boxes." A quasi-minimal cover can be generated via the A^Q algorithm, which we can only sketch here.

B.2 Generation of Interval Covers

We can make this procedure clear by means of a simple example. For 1-D textures shown in Figure 4.1a, the $F^{1\beta}$ and $F^{0\beta}$ are shown in Table 1. For this example, an interval cover (a minimal one, as it turns out) can be generated manually by means of a visual aid, the Generalized Logic Diagram (GLD), which was introduced by Michalski [1] (Figure 4.2).

In the particular case when variables assume only two values, the GLD reduces to a diagram which resembles a Marquand-Veitch diagram. The GLD is a representation of the entire event space, E : 64 events in this case. To use it, the events of $F^{1\beta}$ are mapped in as ones and those of $F^{0\beta}$ as zeros. The squares left over represent F^* (don't cares). The cover is found by an iterative procedure which begins by picking the first "one" encountered in a TV-like scan of the GLD, and discovering all of the maximal intervals which include that "one," but no zeros (an interval "star"). One of these, the interval including the most "ones," is added to the covering set (initially empty). All of the ones included in the "star" are temporarily eliminated, and the scan of the GLD is resumed. The first "one" encountered is selected, and the iteration repeats. Eventually all the "ones" have been eliminated. If all "ones" are included in the covering set, then the cover is minimal. Otherwise, the cover is patched up to include the neglected events, and may not be minimal.

This procedure was followed for the example, and a minimal covering using three intervals resulted: (Figure 4.1b)

$$L_1 = \overset{2}{x}_2^3 \overset{3}{x}_3^3, \quad L_2 = \overset{1}{x}_1^3 \overset{0}{x}_2^2 \overset{2}{x}_3^3, \quad L_3 = \overset{2}{x}_1^3 \overset{2}{x}_2^3$$

If these are used to form a categorizer Ψ_R , where $\Psi_R(e) = 1$ for $e \in R = L_1 \cup L_2 \cup L_3$, then the event categorization shown in Figure 4.1 results. The asterisks appearing above Texture 1 and Texture 0 indicate events for which Ψ_R had output = 1. The subscripts on the asterisks denote the interval producing the "hit." Notice that the probability of a hit ($19/20 = .95$) and the probability of a false alarm ($3/20 = .15$) is as predicted by the ROC (Figure 4.1c) for the likelihood ratio decision rule with $\beta = 1$. These textures could easily be discriminated by labeling regions with hit density over some averaging aperture greater

than, say, 55% as Texture 1. In a digital parallel processor, like the Illiac III's Pattern Articulation Unit, Ψ is an image filter. The input to the filter is a digitized picture in several gray values, and the output is a binary plane labeling each element in the input picture as to which texture the picture element most likely belongs. The application of simple smoothing or noise-removal algorithms would then make segmentation into texture regions relatively easy.

B.3 Intervals as Textural Feature Detectors

Intervals were achieved as "boxes" into which events from the acceptance set R were efficiently packed. As the events in the acceptance set occur more frequently in T^1 than in T^0 , the intervals which are nothing but groups of such events, have a tendency to define features which are more likely to be found in T^1 than in T^0 . Sometimes it was found in a practical case, some intervals pick up features like vertical lines, horizontal lines, herringbone patterns, etc. which are perceived by human beings. Some of the features extracted by other choices of intervals may be very poorly matched for human perception. Strategies for the choice of "good" intervals remains to date largely unexplored.

These "intervals" can be treated as 2-D filters which detect textural features more common to T^1 than T^0 . The normalized "output count" of the filters (normalized number of input "local patterns" from the scene of analysis that fall in the "pass band" of the filter) can be used as a feature vector. In this multidimensional space, we will be dealing with clusters of the scenes from T^1 and T^0 and for classification purposes we can resort to any popular cluster analysis methods.

It may be noted that the generation of "interval covers" was possible for the binary case only. Extension of this procedure for the case of multiple textures is reserved for further investigation.

References

- [1] Michalski, R. S. and B. H. McCormick, "Interval Generalization of Switching Theory," Proceedings of the Third Annual Houston Conference on Computer Systems Science, Houston, Texas, April, 1971. (See also an extended version in Report No. 442, Department of Computer Science, University of Illinois, Urbana, 1971.
- [2] Michalski, R. S., "A Geometric Model for the Synthesis of Interval Covers," Report 461, Department of Computer Science, University of Illinois, Urbana, 1971.



TEXTURE 1

















TEXTURE 0



Figure 5 One Dimensional Texture

Table 1

Event (e)	n_1 (e)	$p(e, T_1)$	n_0 (e)	$p(e, T_0)$	LR(e)
 (1,2,3)	3	.15	0	0	-
 (2,3,2)	5	.25	0	0	-
 (2,1,2)	2	.10	0	0	-
 (1,0,2)	1	.05	0	0	-
 (0,2,3)	2	.10	0	0	-
 (3,2,0)	1	.05	0	0	-
 (2,0,2)	1	.05	0	0	-
 (3,2,1)	4	.20	3	.15	1.33
 (2,1,0)	1	.05	4	.20	.25
 (1,3,2)	0	0	3	.15	0
 (1,0,1)	0	0	4	.20	0
 (0,1,2)	0	0	2	.10	0
 (1,2,1)	0	0	2	.10	0
 (0,1,3)	0	0	2	.10	0

 E^1 E^0

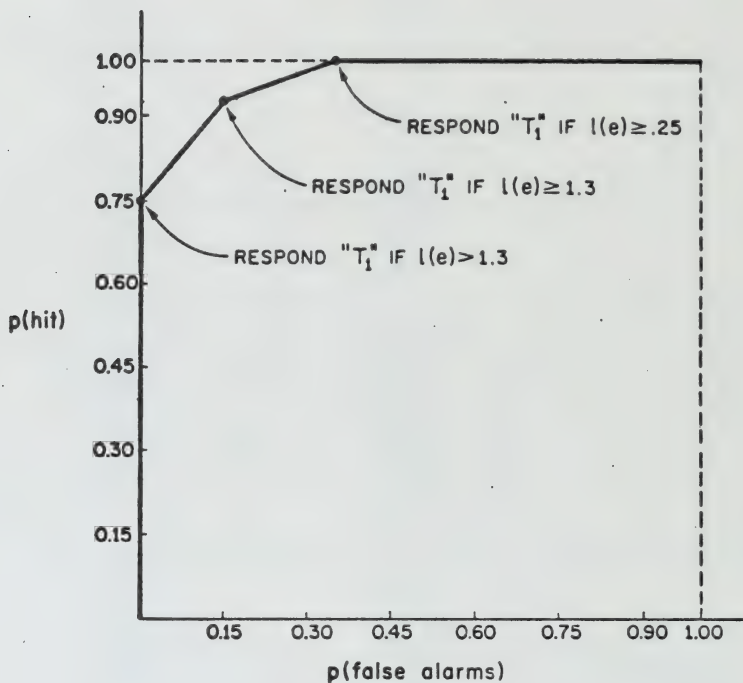
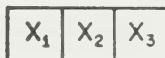


Figure 6. R.O.C. for Textures Shown in Figure 5

TEMPLATE DEFINING EVENTS:



INTERVAL COVER

L_1



$^2X_2^3 X_3^3$

L_2



$^1X_1^3 \text{ } ^0X_2^2 \text{ } ^2X_3^3$

L_3



$^2X_1^3 \text{ } ^2X_2^3$

NOTE: Lower limit for each variable is indicated in lower right hand corner and the upper limit in the upper left hand corner.

Figure 7. Interval Cover for Textures Shown in Figure 5

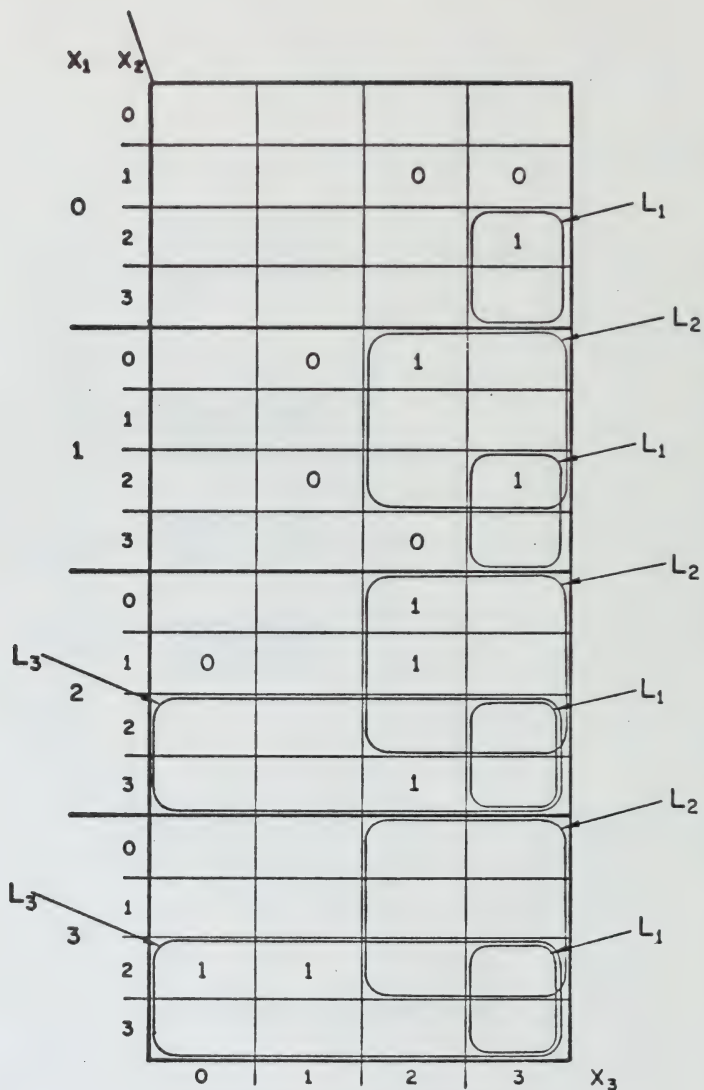


Figure 8. Generalized Logic Diagram with Interval Covering

4.4.4 ATLAS-DEFINED INFORMATION: IMAGE DEFORMATION

A. Eidetic Approach to Picture Representation and Analysis

Linguistic and graph-theoretical methods of representing and analyzing pictures are presently very popular in the field of pictorial pattern recognition. These methods are certainly useful for some applications, especially, e.g., for generation of pictures or analysis of limited classes of pictures such as line-drawings or plane-bounded objects. It seems, however, that for processing complex natural images these approaches may not be sufficient, and some new concepts may be needed. An ideal we have chosen to investigate is based on the assumption that a final representation of a complex image is not a graph or set of sentences in a graphical language, but a labeled pictograph or map which preserves, in a simplified form, basic spatial relationships occurring in the original picture and has references to data banks containing information about elements of the pictograph.

A simple observation which leads to such a concept is that, e.g., for a human who wants to get a quick orientation in an unknown terrain or city, even a crude map is by far more useful than a verbal or written description. An approach to picture analysis, description, and (hopefully) recognition based on such a labeled pictograph representation of graphical objects we will call pictographic or eidetic (from the Greek word eidetikos--which means having the characteristics of eide; eidos (pl. eide)--form, shape).

Some of the problems which immediately arise from such an assumption are:

1. algorithmic synthesis of maps for given images,
2. matching images with maps which describe them,
3. investigation of the "best" map representation for a given image.

B. A First System

Given both the map and the image, the operator selects the corresponding areas or windows that he wants to match. As in Figure 9, the map (Film 1) is linearly distorted until the center of the window matches the image (Film 2). The result is shown in Figure 9. If the window is sufficiently small, the map \rightarrow image transformation can be well approximated by a linear transformation, namely, the composition of translation, rotation, skew, and possible rescaling. These functions are either already implemented or can be implemented in the available hardware.

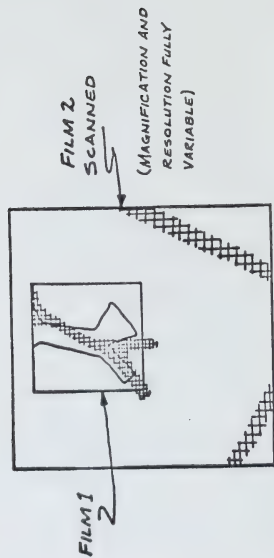
As more windows are matched, a set of linear approximations is obtained. The centers of these approximations form a net lattice which divides a given part of the map into disjoint triangles; an interpolated approximation of the map \rightarrow image transformation is then calculated for each triangle (Figure 10). If necessary, more windows are selected until a satisfactory overlay is obtained. The union of these individual transformations is taken to be the map \rightarrow image transformation, and the map is then transformed and overlaid in the image.

This procedure provides a natural manner of solution if we were to have a very flexible map overlay which we could stretch and compress until it fitted the image beneath. Interpolation functions have been found which should be able to approximate any degree of plastic (topology-preserving) deformation to any specified accuracy. These functions are third or fourth degree polynomials, but since they are applied in triangular patches and

not over the entire area of the map, no great loss of accuracy results. They have also been carefully selected so that the transformations are at least first order continuous across triangle boundaries. Figure 11 gives a typical triangle.

C. Remarks

Besides the basic problem of map \rightarrow image correspondence, there remains the task of implementing the indicated deformations (map/image) economically by low order extrapolation formulae. Algorithms to do this have been suggested by related work on economical curve generation tactics in computer graphics. Currently these techniques are being implemented and evaluated here.



1. MAN SELECTS WINDOW OF MAP HE WANTS TO MATCH WITH IMAGE.

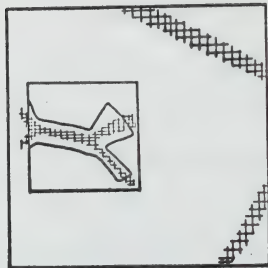
(CURRENTLY, MAP [FILM 1] HAS TO BE RESCANED FOR EACH NEW WINDOW)

2. MAN MATCHES CENTER OF MAP WITH IMAGE USING 'TRANSLATE' AND 'SKEW' OPERATIONS.

A) TRANSLATE ALREADY AVAILABLE AS A CHANGE IN SMV PARAMETERS.

B) SKEW AVAILABLE AS SIMULATED (FOR 32x32x4, 4ms COMPUTATION < 1ms 1/6 TIME).

Image Overlaid with Map (Misaligned)



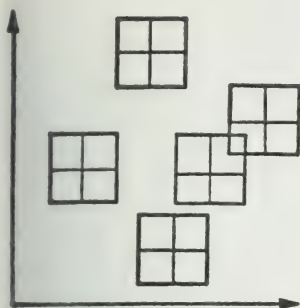
AFTER SKEW AND TRANSLATE
(CENTER OF MAP MATCHES IMAGE,
EDGES NEED NOT)

1. SIZE OF MAP WINDOW CHOSEN SHOULD REFLECT CONFIDENCE OF MATCH- THE LARGER, THE BETTER.

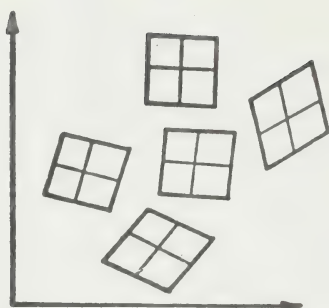
2. CURRENTLY, HUMANS DO THE MATCHING- LATER, WE MAY BE ABLE TO HAVE THE MACHINE DO IT.

Image Overlaid with Map (Aligned)

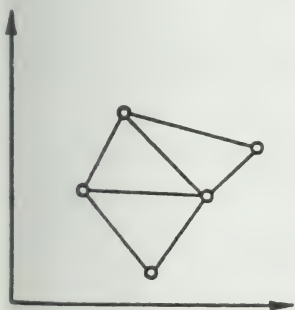
Figure 9



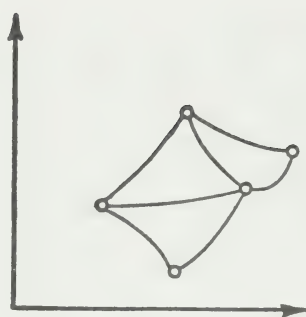
WINDOWS SELECTED IN MAP
(CENTER LINES REFLECT SKEW)



RESULTANT WINDOWS
AFTER MATCHING



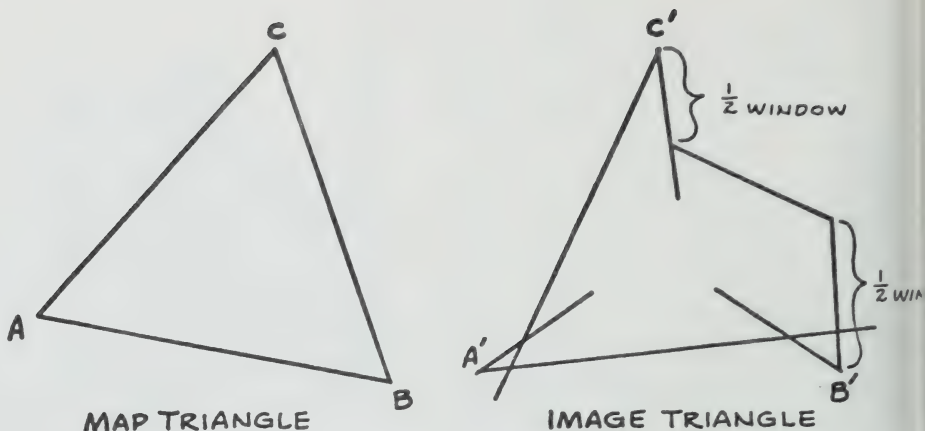
TRIANGULARIZATION
OF MAP



RESULTANT

1. TRIANGULARIZATION IS USED TO COMPUTE TRANSFORMATION OF EDGES. THEN INTERPOLATION IS USED TO FIND TRANSFORMATION OF INTERIORS OF TRIANGLES.
2. THIS PROCESS CAN BE ITERATED UNTIL A SATISFACTORY MATCH OF THE ENTIRE MAP IS OBTAINED.
3. CURRENTLY, TRANSFORMATION IS DONE IN SOFTWARE. THESE PATCH-TRANSFORMS ARE CONTINUOUS ACROSS EDGES (0^{TH} AND 1^{ST} ORDER).

Figure 10. Transformation Defined by Triangularization of Map



1. MAP TRIANGLE WITH ANGLES A, B, C BECOMES IMAGE WITH ROTATED ANGLES A', B', C' .
2. BEZIER POLYNOMIALS OF DEGREE 2 OR 3 ARE USED AS A SMOOTHED REPRESENTATION OF EDGES OF AN IMAGE TRIANGLE.

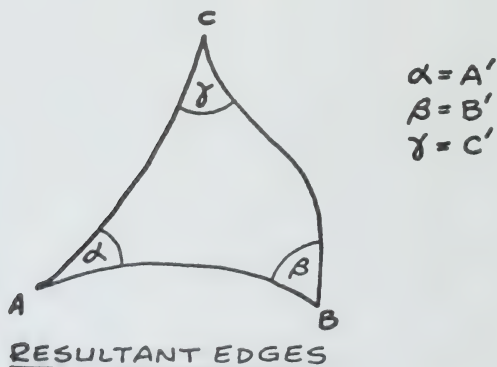


Figure 11. Illustration of Edge Transform Calculation

4.4.5 SHAPE IDENTIFICATION

Stimulated by the publication of Wertheimer's "Principles of Perceptual Organization" in 1923, many psychologists have studied intensively human perception of shape and form. A good survey paper of these studies is found in Hake (1957), and a recent book, Visual Perception of Form by Zusne (1970), is an excellent summary and contains an extensive bibliography. A 1966 collection of papers edited by Leonard Uhr deals with various problems of pattern recognition by computers. However, many of the more crucial problems such as how humans encode shape information, detect near similarity or dissimilarity of shapes, focus attention upon particular aspects of shapes, and achieve perceptual invariance are not yet adequately understood.

A basic weakness of shape/form recognition by present computer systems may stem from the observation that shape/form recognition by humans has not yet been solved on the psychological level. Zusne (1970) says:

In the history of modeling pattern perception on the computer, the task is really one of modeling discrimination rather than recognition. Recognition in a computer is recognition only in a trivial sense. Recognition by organisms implies previous experience, and with experience there accrue meanings to patterns that are uncorrelated in any important way with the physical dimensions of patterns. Whether a machine will ever be able to cope with this problem is uncertain. (page 84)

The objectives of the thesis of Maruyama* are to investigate some intrinsic properties in shapes and to provide some good ways to describe objects in such a way that the high level analysis of pictures can be done easily. Furthermore, we hope that this study provides means to construct a computer system in which the system responses are demonstrably similar to an average

* K. Maruyama, "Shape Detection," DCS thesis in progress.

human response on a specified task¹ and to provide clues for exploring some unsolved problems of shape perception in psychology.

The thesis of Maruyama, to be completed before the end of the present contract period, treats shape analysis under eight chapters:

1. Introduction
2. Topological Discussion (Jordan curves, etc.)
3. Representation of Objects (angularly simple regions, etc.)
4. Problems in Two Dimensions (sofa problems, path finding, etc.)
5. Feature Extraction (description and efficacy of selected shape parameters)
6. Regions and Graphs (analysis of composite figures)
7. Problems in Complex Figures (regional matching)
8. Conclusions and Suggestions

A common quest running throughout this work is to identify and test algorithms which would make possible the automated use of map information to interpret pictorial information.

¹ task: detection, discrimination, recognition, identification, judgment
(Hake 1957)

4.4.6 STRUCTURE TRANSFORMATIONS

The main results of the thesis of Schwebel* are: the definition of a graph-structure transformation model, the development of formal types of embedding properties of a relation under a transformation, and the definition of a graph-structure operation language.

The graph-structure model provides a framework in which to express multi-relational scene segmentation and structural inference techniques. The graph-structure representation and transformations are defined as having properties to facilitate their application to the parsing of pictures.

Simple embedding concepts of relations have been considerably expanded so that relations among submembers of composite structures can infer relations between the composite structures. Picture processing applications of embedding types have been demonstrated.

A structure operation language, SOL, has been defined to provide a means of experimenting with heuristic structure transformation procedures; in particular, SOL is intended to provide an implementation of the graph-structure model.

*J. C. Schwebel, "A Graph-structure Transformation Model for Picture Parsing," Ph.D. thesis, University of Illinois at Urbana, Champaign, June, 1972.

4.4.7 EXPERIMENTS IN IMAGE PARSING

We are now building a small inventory of simple scenes (in first instance from children's coloring books) as a test environment for evaluating picture parsing strategies. With the graph structure transformation concepts in mind, these scenes (~100 in number) are being hand analyzed, prior to encoding their analysis using the SOL language.

In particular this experience with simulated picture analysis has suggested notable extensions of the image representations. Increasingly we find ourselves using crude 3-dimensional models of the more frequent items in the environment, which then we project and attempt to match to the projected 2-dimensional image presented us. These objects need not be rigid (wings flap, arms move, etc.). Accordingly we find that the map \rightarrow image correspondence techniques are also entering this area of picture parsing.

4.5. APPLICATIONS

4.5.1 IMAGE PROCESSING IN AUTOMATED CYTOLOGY

A. Introduction

Images of biological cells are being increasingly used in quantitative studies of cell properties. New techniques have become available which elucidate subtle cell structures and functions. Procedures such as stoichiometric staining, fluorescent or radioactive tagging with antibodies or chemical precursors, and autoradiography produce images whose optical properties can be used to analyze complex biological events. However, conversion of optical properties to numeric form and subsequent interpretation remain problems, especially when large cell populations are involved. Information in biological images is typically rather "noisy," owing to the heterogeneity and variation inherent in such materials. Simple automatic approaches to data extraction are sensitive to this noise. Frequently, morphological considerations are needed to limit the domain of measurement to particular structures to avoid spurious signals [1, 2, 3, 4]. Data conversion and interpretation by humans, on the other hand, can cope with the noise problem, but are not cost-effective for some applications of interest, even when augmented by interactive computation facilities.

What is needed, then, is the development of cost-effective image-processing algorithms which can cope with morphological criteria at the level of complexity present in biological materials. These algorithms can produce some measurements directly, or can be used in conjunction with other analytical instruments. For example, an image-processing algorithm could be used to locate a cell nucleus, whereupon an electron microprobe could be automatically directed

to perform a chemical analysis. Conversely, image-processing could be used to automatically apply quantitative adaptations of human-oriented morphological criteria to the output of existing high-speed cell analyzers. Flow and electrostatic particle transport systems have the capacity to handle, analyze and even sort large numbers of cells, often as many as 100,000 per minute. These devices typically apply measurements, such as light absorption at a particular wavelength, to an entire cell at once with no attempt to resolve intracellular structure. Some applications would benefit from the ability to rapidly filter large quantities of cells using high-speed whole-object measurements followed by automatic morphological analysis of the residue.

B. Review of Cytology Automation

In the thesis of John Read* a particular application of digital image processing in cytology is examined: automated analysis of the well-known "Pap" smear. Pap smears are samples of epithelial (skin) cells from the uterine cervix and are used to detect cancer of the uterus while it is in an early stage and relatively easy to cure. Development of an automatic device to screen these samples has been recognized as of great potential value, almost from the time the test was first devised by Papanicolaou. However, this has proven to be a formidable task, as demonstrated by unsuccessful development projects undertaken by such organizations as the National Cancer Institute, IBM Corporation, and Vickers, Ltd. Accordingly, the goals of this thesis are prudently restricted to something less than a complete implementation. Actual implementation of a system to analyze cervical smears would require a careful analysis of the options available in staining, specimen preparation, transport

*John S. Read, "Parallel Image-Processing for Automated Cervical Smear Analysis," M.C. Thesis, Department of Computer Science, Report No. 497, University of Illinois, June, 1972. Supported in part by AEC Contract AT(11-1)-2118. (This thesis has a bibliography of 271 references of cytology automation.)

and sensing to determine the most effective combination. It seems likely that much deeper insights into the nature of cancer cells will be forthcoming in the next few years. One would expect that this new knowledge will have a significant effect on the technological direction.

C. Image Characteristics of Cervical Smears

In the present work, the prime objective was to examine the effectiveness of a parallel digital image processor in the analysis of cervical smear imagery. A wide range of activities could support this purpose. In order to define the problem adequately for machine implementation, and also to avoid expending excessive time on a complex biomedical problem, it was decided to accept as correct certain conclusions drawn in the course of development of the Cytoanalyzer project, an early attempt to automate the screening of cervical smears by image processing. The pertinent conclusions, paraphrased from references [5] and [6] are as follows:

(1) Normal cells shed from the cervix and vagina exhibit a functional relationship between size and optical density of the nucleus that approximates $e(n) = Q/D(n)^2$ where $e(n)$ is nuclear optical density and $D(n)$ is nuclear diameter suggesting that the nucleus contains a constant quantity, Q , of dye-binding material in normal cells throughout various stages of differentiation.

(2) There is a continuous spectrum of change toward increased optical density of the nucleus and increased nuclear size (relative to overall cell size) as one progresses from normal cells to cancer cells.

(3) Smears classified as being associated with cancer show the presence of a second population of aberrant cells superimposed on the normal cell population.

As is described in the thesis of Read, a prototype instrument incorporating these principles was built which was intended to measure nucleus optical density and nucleus diameter of epithelial cells. Due to limitations in the image processing technology then available, it proved to be impossible to meet the conditions of the qualifying statement. The prototype could not distinguish enlarged, dark, epithelial nuclei from clumps of white blood cells nor distinguish white blood cells from certain normal epithelial cell nuclei. Thus, the validity of the conclusions reached in the design study could not really be tested in a clinical environment because of the presence of this biological image "noise."

D. Cytological Image Processing Experiments

The experiments reported in this thesis were motivated by the assumption that improved image processing technology, as represented by a parallel digital image processing device will permit reliable analysis of much more complex patterns of cell images. The emphasis here is not on the discovery and extraction of parameters for distinguishing a malignant cell from a non-malignant cell, but rather on the construction of algorithms for a parallel processor which permit the machine to rapidly make sense out of the mess of cells and debris in the microscope field so that subsequent measurements (whatever they might be) are made on the correct objects: epithelial cell nuclei, for example, rather than clumps of white blood cells, cytoplasmic folds, or other locally similar-appearing phenomena.

The interest in algorithms for a parallel image processor stems from the belief that for this and similar applications, parallel digital image processing has the greatest likelihood of being sufficiently fast, flexible and cost effective to perform the critical initial steps of object location and identification.

The remainder of this thesis consists of two parts. The first, Chapters II and III, contains background information about the application and a review of the rather extensive literature on automated cytology. The second part, Chapters IV and V, describes some experiments in applying parallel image-processing to some of the cervical image difficulties mentioned above. The objective was to write and test programs for hardware like the Pattern Articulation Unit of Illiac III, where the programs could cope with three particular aspects of the analysis of cervical images:

(1) Rapid filtering of images of minimal photometric and spatial resolution to detect dark blob-like regions representing potential malignant cell nuclei. The Cytoanalyzer study mentioned above provides the justification for searching for large, symmetric dark regions. The constraint to use images of low resolution is in keeping with the realities of scanner performance in situations where high speed is required. The filtering procedure is designed to distinguish between dark blobs caused by symmetric clumps of leukocytes and blobs caused by other objects, including malignant cells.

(2) Detection and counting of normal, well-differentiated epithelial cells. This is again a filtering operation to be done on a low-resolution image, and is useful in cervical smear analysis as an indication of cell sample adequacy and as an input to a stopping rule.

(3) Textural discrimination between cell nuclei and blobs caused by drying artifacts. After the blob filter isolates areas of potential interest, blobs caused by spurious phenomena such as drying artifact can be eliminated by texture or spectral analysis on a higher resolution image. Data rate requirements are much reduced by rescanning at higher resolution only areas which require higher resolution to clear up ambiguities.

While there are many other aspects of cervical cell image analysis which could be of interest, it is felt that these three are particularly compelling, since previous attempts to handle them were unsuccessful because of the state of image processing technology. By capitalizing on the recently-available ability to make better use of two-dimensional information at high speeds, it is felt that these difficulties are solvable.

References

- [1] Ladinsky, J. L., Sario G. E., and Peckham, B. M., "Cell Size Distribution patterns as a Means of Uterine Cancer Detection," Journal of Laboratory Clinical Medicine, Vol. 64, p. 970, 1964.
- [2] Mawdesley-Thomas, L. E. and Healey, P., "Automated Analysis of Cellular Change in Histological Sections," Science, Vol. 163, p. 1200, 1969.
- [3] Spencer, C. C. and Bostrom R. C., "Performance of the Cytoanalyzer in Recent Clinical Trials," Journal of the National Cancer Institute, Vol. 29, No. 2, 1962.
- [4] Husain, O. A. N., and Henderson, M., "Observations on the Use of the Quantimet Image Analysing Computer in Automatic Scanning for Malignant Cells," in Cytology Automation, D. M. D. Evans (ed.), pp. 214-27, Livingstone Ltd., Edinburgh, 1970.
- [5] Tolles, W. E., Horvath, W. J., and Bostrom, R. C., "A Study of the Quantitative Characteristics of Exfoliated Cells From the Female Genital Tract: I. Measurement Methods and Results," Cancer, Vol. 14, No. 3, 1961.
- [6] Tolles, W. E., Horvath, W. J., and Bostrom, R. C., "A Study of the Quantitative Characteristics of Exfoliated Cells From the Female Genital Tract: II. Suitability of Quantitative Cytological Measurements for Automatic Prescreening," Cancer, Vol. 14, No. 3, 1961.

4.5.2 BRAIN MAPPING

In research concerning the study of neuro-interconnectivity relations, a primary problem has been the vast amount of time needed to get meaningful data from the counting of neural cell slides. This operation involves the occupation of trained individuals working with a microscope at oil immersion power.

The purpose of research this past year has been to study the feasibility of using the Illiac III system to scan, count, and identify neurons from biological slides of cat brain limbic system. If such a project proves to be feasible (experiments are still in progress and will be reported in the thesis of Jaecklein), then an attempt will be made to use the system to perform intercomparative data-gathering operations on actual brain slice slides whose neuron count has already been determined. A comparison of the machine count and the manual count will then be made, using the manual count as the standard for accuracy.

A preliminary study was also made of a file structure appropriate for encoding Golgi-stained neural tissue. A key problem here is to simulate in the data base the natural connectivity of the locally identified fragments, e.g., one must be able to recognize that two axonal segments are neighboring and yet emanate from distal and different soma. A report on these experiments is in progress.

4.5.3 CYTOSPECTROMETER

A new and fundamental technology for the identification and sorting of cells and subcellular components has been suggested. A principal ingredient of this technology is the transport of charged particles (cells, subcellular components or macromolecules) by an iterative system of electrostatic quadrupole lenses. These techniques, though extensively used in accelerator installations for the transport of nuclear and high energy particle beams, have not had comparable application to the guidance, identification and separation of macromolecular and cellular particle beams. It was our proposal this past year to explore this potential and to devise a first working instrument (called here a "cytospectrometer") which can bridge the gap between these digitally-oriented beam techniques and more conventional methods in cytology emphasizing light and electron microscopy.

A. Objective of a Cytospectrometer

To provide a 2-dimensional spatial separation of constituent particles (cells, subcellular components or macromolecules) on the basis of a 2-parameter differentiation of the incident stream of isolated particle is the objective. Parameters available for selection include mass, charge, shape, differential spectral absorption/scattering, rate of discharge in an ionizing atmosphere, etc. The procedures used ideally should maintain the identity of subpopulations, such as all chromosomes from a given nucleus.

B. Goals and Aims

Applications of the cell beam technology that warrant investigation include:

- i) Applications to hematology (differential counts, etc.)

- ii) Pap smear classification
- iii) Separation and classification of bacteria
- iv) Isolation of genetic mutants
- v) Subcellular particle separation: e.g., cell nuclei differentiation
- vi) Karyotyping of chromosomes
- vii) Cancer cell discrimination, including possibly stem cell identification.

C. The Basic Instrument

Constituent subassemblies of a cytospectrometer facility (figure) include the following:

- 1) Sample preparation facilities (preparative zonal ultracentrifuge, etc.), for the isolation of cell populations and the fractionation of subcellular components.
- 2) Injection assembly, a digitally controlled hypodermic syringe, to inject a liquid jet of particles suspended in a solute carrier. Rayleigh instability of the jet is used to form a synchronous stream of droplets; a nozzle-skimmer of Kantrowitz-Grey design then strips off the solute--leaving the particle stream.
- 3) Quadrupole optics, to guide the charged beam of particles (cells, subcellular components or macromolecules). Note that the beam transport takes place in vacuum to ameliorate viscous drag.
- 4) Particle detection electronics, to monitor the particle trajectories and extract appropriate identification parameters in flight (spectral absorption, time-of-flight, etc.).
- 5) Two-axis electrostatic deflection system to prepare a 2-dimensional particle spectrogram on a glass slide.

- 6) An automated light microscope for optical scanning both of the gross particle distribution and the individual particles (cells, nuclei, etc.) of the spectrogram.
- 7) Automated on-line scanning electron microscope (SEM) for examination of individual particles of the spectrogram.
- 8) Multivariate classification procedures, clustering techniques and graphical displays to assist in the interpretation of the particle spectrograms.
- 9) Where warranted, facilities to recycle cells through the system, possibly by growing cell colonies from individual cells of the distribution.

D. Achievements

1) A prototype cytospectrometer was constructed of modular vertical cylindrical sections. The section to house the liquid cone is, perhaps, more important and was fitted with a long working distance microscope objective and viewing assembly.

2) The thermal pump scheme was used to see if a Taylor cone could be formed using water. Previous attempts at cone formation with static water levels at the needle tip met with near zero success because there was no liquid flow to replace material drawn off at the tip. This doubtless led to collapse of the cone in the absence of increasing voltage. Hence the pump was added and the test run again. This test was somewhat more successful as the cone (or something like it) formed and remained "stable" for perhaps a second at a time before lapsing into wild instability--followed by periodic formation and collapse as the pump caught up with the flow demand and then fell behind. Tests are continuing.

3) Initial helical quadrupole beam transport tubes have been fabricated, using 4 mm OD glass tubes. Evaluation of the focusing action of these assemblies using a droplet generator and strobed light source is now in progress.

4) An integrated circuit control unit for droplet generation, pulsed displacement and strobed display has been completed and is operational. This unit, using digital logic throughout, has seen extensive use in our test evaluation of cytospectrometer operation.

The cytospectrometer project has run on approximately \$7000 in equipment and 0.5 FTE personnel investment this past year.

4.6. COMPUTER SYSTEMS SUPPORT

4.6.1 IBAL ASSEMBLER

IBAL is the Illiac III Basic Assembly Language. Like other assembly languages, with suitable constraints upon the naming of operands, each primitive operation in IBAL directly corresponds to one Illiac III machine instruction. It differs, however, from conventional assembly languages in its ability to structure data more generally and more conveniently. This feature facilitates and unifies the implementation of higher level complicated data structures which are predicted in the application course of the Illiac III computer system. This attention to data structure also simplifies the task of application system programmers by making the detail of operand addressing invisible to them, which is otherwise a rather difficult programming task.

A statement in IBAL may be a block, declaration, instruction, or directive. Blocks permit the specification of nested structures of statements and automatic data allocation. Declarations allow specification of generalized tree-structured data sets whose irreducible constituents are the smallest addressable basic elements of the machine. The complete specification of the language is given in [1].

In the course of implementing the language, we have tried a few compiler-compiler systems. Bottom-up compilers initially looked very attractive due to the fact that we could incorporate very sophisticated error recovery procedures, and, in the course of our experimentation, we generated Floyd productions for the syntax specified in [1]. Unfortunately, the system we used (TWS) was not adequately documented, and it proved

extremely difficult to modify it for the handling of our compiler. Next we tried and generated a top-down parser, which involved a long algol program.

Currently we are implementing a two-pass IBAL translator, where the first pass mainly deals with syntax and data declarations and produces an intermediate code to be processed by the second pass. As the specification of symbol tables and the actual semantic compilation structures have been laid out, we have noticed that the major part of the translation is carried out in the first pass, and the intermediate code generated by the first pass has an almost one-to-one correspondence with machine instructions, which makes the second pass rather trivial. There remains the fact that a sophisticated Pointer Register (PR) Allocation algorithm (yet to be developed) must be incorporated in the actual machine code generation, which is believed to be a key factor for execution time efficiency. The symbol tables and general structure of the translator have been defined and will be documented and published before the end of the present contract period.

Another major point to be studied yet is the interface between the assembler and the operating system of Illiac III. The following points have been resolved in our implementation:

- (1) Memory allocation
- (2) Procedure prelog and epilog
- (3) Calling sequence (convention)
- (4) Actual formal parameter mapping
- (5) Relation between the active display and data area
- (6) Run time storage for structures

In summary, the parsing phase of the language is near completion. The most incomplete part of the language is the semantic phase. Finalization of this language is essential for the preparation of diagnostic tests for Illiac III.

References

- [1] John C. Schwebel, "IBAL Manual: The Illiac III Basic Assembler Language," revised by Ahmad E. Masumi, DCS Report No. 469, University of Illinois at Champaign-Urbana, July, 1971.

4.6.2 PATTERN ARTICULATION UNIT DEVELOPMENT

The Pattern Articulation Unit employs a two-dimensional Iterative Array (IA) of 1024 (32×32) identical processing modules locally connected to execute Boolean functions, threshold logic, and signal path building. It is augmented by its own control unit and by an unconventional memory, called the Transfer Memory (TM), which, in conjunction with the Iterative Array, can operate as an associative memory.

Together the IA and the TM provide a stack of planes for plane parallel picture processing. In this context the IA can be considered a set of fast scratch-pad registers (of "word-length" = 1024 bits) for the supplementary TM store (again of 1024 -bit planes). One half of the IA is currently operating, with about 75 per cent of the control, and is running diagnostics.

The PAU developments of the past year will be described according to the logical subdivisions of the PAU.

* Control: The principal addition to the PAU control was the "back door" interface to the maintenance processor (PDP8/e). This interface makes the maintenance processor look like a word in the Read Only Memory. Thus, gate commands may be sent directly to the Iterative Array. This facility is useful in two ways: 1) it allows checkout of gate sequences for PAU instructions before they are implemented, and 2) it allows maintenance programs to exercise the IA.

Two software packages have been developed to use this path. The first is an Iterative Array simulator which runs on the PDP8/e. The simulator can perform all the IA functions; in addition it can print out the

contents of planes as well as print out both the logical coordinates of points of interest and the physical hardware location of such points.

The second software package is a simple interpreter which allows the user to build up IA gate sequences in a quasi-mnemonic form. It includes unconditional branches and looping and has proved extremely useful in performing diagnostics on the IA.

The border instructions (viz. LOADB, STOREB, PUSHB, and POPB) have been designed and implemented. They currently operate on the NORTH border with one bit of gray scale.

The GATEIA and TOPOLOGY instructions have been running without a failure.

* Iterative Array: During the last year, all inter-bay neighbor wiring remaining to be done is the inter-section wiring between the two halves of the array.

Wiring was also added for both the control and the data paths for the border instructions.

Power distribution was also completed for sections 5B (IA drivers in control for section 3) and 3T (the remaining quarter of the IA).

* Display system: The display system has been running for the last year without failure.

* Transfer Memory: The logical design of the PRE of TM II was completed. Bid specifications are being written for the fabrication and wiring of the PRE drawers.

The discrete component cards for interfacing with the Iterative Array have been constructed and are undergoing prototype evaluation.

The scratchpad memory chips have been ordered and the memory card design is about 95% complete. There are eight such cards in the system; each contains 397 chips and about 2400 wires.

4.6.3 ARITHMETIC UNIT DEVELOPMENT

During the past year the design and fabrication of the Arithmetic Unit was completed. Specifically, all the discrete component cards (processing hardware) were tested; the integrated circuit control cards were designed, fabricated, and tested; and the back panel wiring was completed.

Various strategies, including use of the maintenance processor, are being evaluated to perform the entire system checkout.

4.6.4 TAXICRINIC PROCESSOR DEVELOPMENT

The fabrication of the TP processing hardware (discrete component cards) was completed. All the cards have been tested, and all back panel wiring changes have been made.

The logical design of the basic machine control (184 integrated circuit cards) was completed, and approximately 75 per cent of the cards were wired and tested. About 25 per cent of the necessary back panel wires for the basic control have been installed.

The logical design of the final machine control (160 integrated circuit cards) is about 50% complete (June 20, 1972).

4.6.5 PROCESSOR INTERCOMMUNICATION

The Exchange Net, the vehicle for processor inter-communication in the Illiac III system, was completed in October. It has been used extensively, without failure, by the Show-and-Tell system since then.

4.6.6 SCANNER/MONITOR USAGE

The scanner/monitor usage for the past year breaks down as follows:

Production and demonstration	643 hours 55 minutes
Preventative maintenance	84 hours 50 minutes
Corrective maintenance	194 hours 40 minutes

The failures rectified by the corrective maintenance were:

- 1) A wiring error whose effects were visible only at very high resolution when looking at very thin lines.
- 2) A bad transistor on an over/under voltage regulation card in the power turn-on system.
- 3) A bad Dormeyer relay in the local power regulation box.
- 4) Five fuse failures in the Modine regulators for local power distribution.
- 5) A monitor high voltage supply which lost regulation.

4. 7. BIBLIOGRAPHY

4.7.1 EXTERNAL DOCUMENTS ISSUED

A. Outside Lectures

B. H. McCormick, Washington University, St. Louis, "Seeing and Believing," September 10, 1971.

John Read, Two Dimensional Digital Processing Conference, University of Missouri, Columbia, "Problems and Programs for Computer Processing of Microscope Images," October 6-8, 1971.

B. H. McCormick, Annual Review of Electronics Conference. University of Illinois, Urbana, "Digital Systems for Automated Cytology, October 18, 1971.

B. H. McCormick, Distinguished Visitors Program of the IEEE Computer Society, University of Missouri, Columbia, Missouri, "Image Processing in Automated Cytology," November 11, 1971.

R. S. Michalski: Lectures on Variable-valued Logic and Its Applications to Pattern Recognition, Picture Processing and Artificial Intelligence.

February 7, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada

February 9, Department of Applied Analysis and Computer Science, University of Waterloo, Ontario, Canada

February 10, Department of Electrical Engineering, McGill University, Montreal, Canada

February 14, Project MAC, Massachusetts Institute of Technology, Cambridge, Massachusetts

February 16, Center for Information Research, University of Florida, Gainesville

May 26, Department of Computer Science, Oregon State University, Corvallis

May 29, Department of Computer Science, Stanford University, Palo Alto

B. H. McCormick, IEEE Information Theory Group Image Processing Workshop, University of Notre Dame, Indiana, "Economy of Description in Image Processing," May 5, 1972.

B. Seminars: "Pattern Recognition in Biological and Technical Systems"

C. W. Eriksen, "Selective Attention in Visual Perception," October 12, 1971.

John Read, "Problems and Programs for Computer Processing of Microscope Images," October 21, 1971.

Kiyoshi Maruyama, "Some Problems of Modeling of Human Shape Perception," October 28, 1971.

Heinz Von Foerster, "Three Problems of Computation in the Central Nervous System," November 3, 1971.

Charles Lewis, "Perception of Depth in Pictures," November 10, 1971.

L. E. Jones, "Multiple Discriminant and Regression Models for the Analysis and Representation of Multidimensional Psychophysical Relationships," November 17, 1971.

M. L. Babcock, "Asynchronous Sampling of Speech," December 1, 1971.

Emanuel Donchin, "Computer Aided Studies of Event-Related Brain Potentials," December 9, 1971.

Peter Shaw, "Pattern Recognition by Humans of Letters and Words," December 15, 1971.

C. Department of Computer Science Reports:

Report No. 469	Schwebel, J. C., Revised by Masumi, A. E., "IBAL MANUAL - The Illiac III Basic Assembler Language," July, 1971.
UIUCDCS-R-71-470	Nordmann, B. J., Jr., "Speech Display Simulation System for A Comparative Study of Some Visual Speech Displays," August, 1971.
UIUCDCS-R-71-472	McCormick, B. H., Nordmann, B. J., Jr., et al, "ILLIAC III REFERENCE MANUAL - VOLUME IV: Supervisor Organization," August 12, 1971.
UIUCDCS-R-71-473	McCormick, B. H., Nordmann, B. J., Jr., et al, "ILLIAC III REFERENCE MANUAL - VOLUME III: Input/Output," August 13, 1971.
UIUCDCS-R-71-475	Nordmann, B. J., Jr., "ILLIAC III COMPUTER SYSTEM MANUAL - Taxicrinic Processor, Volume 2," August 24, 1971.

UIUCDCS-R-71-476	Nordmann, B. J., Jr., "ILLIAC III COMPUTER SYSTEM MANUAL - Taxicrinic Processor, Volume 3," August 24, 1971.
UIUCDCS-R-71-479	Nordmann, B. J., Jr., "A Comparative Study of Some Visual Speech Displays," September 10, 1971 (Ph.D. Thesis).
UIUCDCS-R-71-489	Maruyama, Kiyoshi, "An Approximation Method for Solving the SOFA Problem," December, 1971.
UIUCDCS-R-71-490	Maruyama, Kiyoshi, "A Problem in Form Perception: Odd Shape Detection," December, 1971.
UIUCDCS-R-72-496	Raulefs, Peter, "Methodological Aspects of Scene Segmentation," August, 1972 (M.S. Thesis).
UIUCDCS-R-72-497	Read, John S., "Parallel Image-Processing for Automated Cervical Smear Analysis," June, 1972 (M.S. Thesis).
UIUCDCS-R-72-514	Schwebel, John C., "A Graph-Structure Transformation Model for Picture Parsing," May, 1972 (Ph.D. Thesis).
File Note No. 864	McCormick, B. H. (ed.), "Illiac III Bibliography Condensed Listing - Supplement," July 13, 1971.
File Note No. 868	Lewis, G. T., "IMAGE 8: Illiac Paper Tape Reader Control," August 10, 1971.
File Note No. 871	Read, John S., "IMAGE 8: Microscope Stage and Focus Motor Control," June 30, 1972.

D. Papers Accepted for Outside Publication:

Maruyama, K. "An Approximation Method for Solving the SOFA Problem," will appear in An International Journal of Artificial Intelligence.

Maruyama, K., "A Procedure to Determine Intersections Between Polyhedral Objects," will appear in Issue #3 of the International Journal of Computers and Information Sciences.

Maruyama, K., "On the Parallel Evaluation of Polynomials," accepted for publication in IEEE Transactions on Computers.

J. S. Read, S. N. Jayaramamurthy, "Automatic Generation of Texture Feature Detectors," IEEE Transactions on Computers, July, 1972.

R. S. Michalski, "A Variable-valued Logic System as Applied to Picture Description and Recognition," Proceedings of the IFIP Working Conference on Graphic Languages, May 22-26, 1972, Vancouver, Canada.

4.7.2 LOGIC DRAWINGS ISSUED

The following new logic drawings have been drawn and issued during the past year:

TP Control Logic	110 drawings
AU Control Logic	63 drawings
PAU Control Logic	24 drawings
Power Turn-on Logic	20 drawings
SMV Buffer Control	2 drawings
Image 8 (PDP8/e Interface) Logic	7 drawings

4.7.3 ENGINEERING DRAFTING REPORT

During the past year a total of 1194 drawings, including new logic drawings, drawing changes, layouts, flow-charts, theses, report drawings, proposal drawings, and drawings related to the Opto/mechanical design of the Illiac III project have been processed by the 2118 drafting section.

4.7.4 ADMINISTRATION

Senior Staff

Professor Bruce H. McCormick - Principal Investigator

Assistant Professor R. S. Michalski

Professional Staff

Robert C. Amendola
Richard T. Borovec
John S. Read**

Research Engineering Assistant

S. Paul Krabbe

Electronic Engineering Assistant

Joseph V. Wenta

Digital Computer Technician II

George T. Lewis

Drafting

Stanislavs Zundo

*Terminated at end of first quarter

**Terminated at end of second quarter

Research Assistants

Jerry Chen*
Walter Donovan
Steve Fierce**
Lakshmi Goyal**
S. N. Jayaramamurthy
Ahmad E. Masumi
Kiyoshi Maruyama
Frank Murakawa
Dan Pitt**
Peter Raulefs
Kuo Wen

Secretarial

Mrs. Patti Welch

U.S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TOE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

C00-2118-0037

2. TITLE

2nd Quarterly Progress Report 1972

3. TYPE OF DOCUMENT (Check one):

☒ a. Scientific and technical report

☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

☒ a. AEC's normal announcement and distribution procedures may be followed.

☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.

☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

B. H. McCormick, Professor
and Principal Investigator

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature

B. H. McCormick

Date

June 1972

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

☐ a. AEC patent clearance has been granted by responsible AEC patent group.

☐ b. Report has been sent to responsible AEC patent group for clearance.

☐ c. Patent clearance not required.

(Supported in part by the National Science Foundation under Grant Number US NSF GJ 813.)

5.1 Study of Logical Organization for LSI Implementable Arithmetic Units

This study concerns itself with the development of efficient logical organization of an Arithmetic and Logic Unit (ALU), which is consistent with the demands and constraints of large scale integration. A set of characteristics desirable for the arithmetic unit under investigation have been formulated as follows.

- i) The ALU should be partitionable on a digit basis so that calculations can be performed on a digit by digit basis.
- ii) All the digit processing modules (DPM) should be identical so that a variable word length can be accommodated.
- iii) To avoid fan-out problems in case of large word length operands, the various modules should have limited inter-communication with each other.
- iv) Since purely combinational arrays are too expensive and complex for any reasonable radix and word length, the arithmetic should be performed by a compromise between the purely time sequential algorithm and purely combinational LSI arrays.
- v) Consistent with the variable word length nature of the ALU and the fan-out constraints, the control for coordinating the operation of various digit processing modules to execute the arithmetic algorithm should be as uniformly distributed in the various DPM's as possible. The central global control should be as small as possible and should be independent of the word length of the arithmetic operands.

Research during this quarter centered upon the study of various number systems that could help the digit slice partitioning and expansion to any word length of the ALU. The number systems studied were residue number systems, the negative-base system and the signed-digit redundant number representation. Keeping in view the complexity of the arithmetic algorithms, it has been decided to use a signed-digit representation. Structurally, the pipeline organization on a digit basis seems to be promising and some effort was devoted to the study of various pipeline organizations suggested by various authors in the literature.

Research is continuing on the method of distributing control in the various digit processing modules and finding an optimal balance between the global control and local control in the DPM's.

(L. N. Goyal)

5.2 Implementation of Continued Product Algorithms

During the quarter some aspects related to the function evaluation algorithms based on continued products and sums have been considered. A generalized approach has been initiated, in which evaluation of an elementary function is considered as a process P which can be possibly decomposed into two or more subprocesses P_1, P_2, \dots, P_n , with the following assumptions:

- a set of hardware-implemented operators performing simple operations (e.g., add, shift, low precision comparison, etc.) on operands o_1, o_2, \dots, o_n defined in subprocesses is given;
- each subprocess p_i is, for simplicity, iteratively organized;

- at least one subprocess p_j is independent of all other subprocesses and most of these subprocesses are mutually independent, but related to p_j .

If the basic operation, like add, is sequential by nature, the execution of the processes may be overlapped, using, for example, pipelining of the adder. The objective of such an approach is to obtain the result of a complex process P by overlapped execution of n simpler processes, utilizing with the maximal efficiency available hardware. This approach has been found applicable to the continued product algorithms, where only two subprocesses are identified: the normalization subprocess p_1 and the result evaluation subprocess p_2 such that

$$p_1(k+1) = f_1(p_1(k))$$

and

$$p_2(k+1) = f_2(p_1(k), p_2(k))$$

at every step k .

With the recent developments in integrated circuit technology, it is an open question, worth investigating, as to what should be considered as a basic set of operators and what kind of decomposition can be obtained.

With no definite results, a mixed normalization technique, a combination of a multiplicative and additive normalization, has been considered, in order to reduce or eliminate variable shifting networks and to provide grounds for the carry-save mode of operation, impractical in the present formulation of algorithms based on continued products. A more complete insight in the applicability and practicality is yet to be obtained.

(Milos D. Ercegovac)

5.3 Automatic Evaluation of Some Elementary Functions Using Microprogramming

During this quarter, the continued product algorithms for evaluating certain elementary functions^{(1), (2), (3)} has been simulated using microprogramming techniques. We specifically choose the redundancy formulism proposed in Ref. (3) in this simulation. The computer used is the D-machine⁽⁴⁾ by the Burroughs Corporation which is a 16 bit user-microprogrammable mini-computer with a semiconductor micro-memory (and also conventional magnetic-core memory). The cycle time of the D-machine is 400 ns. One particular useful feature of the D-machine is that it has a multiple-shifting network (called the Barrel Switch) which is crucial in the continued product algorithms.

The main results are as follows:

Elementary Function		Argument range	Average number of micro-instructions required on the D-machine
Division	y/x	$x \text{ and } y \in [1/2, 1)$	210
Multiplication	xy	$x \text{ and } y \in [1/2, 1)$	230
Logarithm	$\ln x$	$1/2 \leq x < 1$	240
Exponential	e^x	$-\ln 2 < x < \ln 2$	250
Square Root	y/\sqrt{x}	$x \text{ and } y \in [1/2, 1)$	230
Sine and Cos	$\sin x$ $\cos x$	$0 \leq x < 2\pi$	280
Tangent or Cotangent	$\tan x$ $\cot x$	$0 \leq x < 2\pi$	460
Arctangent	$\tan^{-1} x$	$0 \leq x < 1$	250

So except for $\tan x$ (or $\cot x$) which takes an extra division time, we can evaluate all these functions in around 100 us. The accuracy of the results is limited to 2^{-15} ($\sim 3 \times 10^{-5}$) because of the short word length

of this computer. However, we can easily extend the range of the arguments to all possible values and improve the accuracy of the results if we have a machine with a longer word length and a floating point arithmetic facility.

Besides these results, for the purpose of improving the speed of the continued product formulations in Ref. (3), several termination algorithms have been studied and simulated with limited success.

(Peter D. Ting)

References and Notes

- (1) J. E. Meggitt, "Pseudo Division and Pseudo Multiplication Processes," IBM Journal of Research and Development 6, 210-226 (April, 1962).
- (2) W. H. Specker, "A Class of Algorithms for $\ln x$, $\exp x$, $\sin x$, $\cos x$, $\tan^{-1} x$, and $\cot^{-1} x$," IEEE Trans. Electronic Computers EC-14, 85-86 (FEB. 1965).
- (3) B. G. DeLugish, "A Class of Algorithms for Automatic Evaluation of Certain Elementary Functions in a Binary Computer," DCL Report No. 399 (June, 1970).
- (4) We wish to thank Professor D. Kuck for the use of the D-machine and its simulator on the B5500 computer.

5.4 Continued Fraction Arithmetic

In this quarter work on generating several functions through the use of the Riccati equation was continued. Present investigation is focused on $y_x = \sin^{-1} x$ but the treatment for many other functions like e^x , $\ln x$, $\tan x$, etc. is very similar.

$$y'_{2m} = k_x (a_{2m} y_{2m} + b_{2m})^2$$

$$y'_{2m+1} + k_x (a_{2m+1} y_{2m+1} + b_{2m+1})^2 = 0 \quad m \geq 0, \quad k_x = \frac{1}{\sqrt{1-x^2}}.$$

with the bilinear transformation,

$$y_n = \frac{1}{q_{n+1} + y_{n+1}}, \text{ the recursion relations are,}$$

$$a_{n+1} = b_n$$

$$b_{n+1} = a_n + b_n q_{n+1}.$$

The above system of differential equations has a solution

$$y_0 = \sin^{-1} x \text{ for } a_0 = 0, b_0 = 1.$$

The initial conditions for these equations, is,

$$y_0 = 0 \text{ at } x = 0.$$

$$y_1 \rightarrow \infty \text{ as } x \rightarrow 0.$$

$$y_n = -(q_n + \frac{1}{q_{n-1}} + \frac{1}{q_{n-2}} + \dots + \frac{1}{q_2}) \text{ at } x = 0.$$

Let this be denoted by t_n , then,

$$t_n = \frac{1}{t_{n-1}} - q_n$$

Let

$$t_n = \frac{d_n}{e_n} \quad \text{then,}$$

$$d_n = e_{n-1} - q_n d_{n-1}$$

$$e_n = d_{n-1}$$

$$d_0 = 0, \quad e_0 = 1.$$

Thus we have an iterative continued fraction algorithm for $\sin^{-1} x$ except for the rules of selection of q_1 at every step. The q_1 are currently chosen to be in the digit set $\{1, 1/2, 1/4\}$ but could belong to any set

consisting of powers of two. With this digit set, the range restriction on x is $x \in [0.3302, 1]$.

Algorithm Arcsin. (AS)

AS-1: $a_0 \leftarrow 0, b_0 \leftarrow 1, d_0 \leftarrow 0, e_0 \leftarrow 1$

$p_0 \leftarrow 0, q_0 \leftarrow 1;$

AS-2: $q_1 \leftarrow \text{select}(x, a_0, b_0, d_0, e_0),$

$p_1 \leftarrow 1, q_1 \leftarrow q_1,$

$q_1 \leftarrow 1, b_1 \leftarrow q_1, d_1 \leftarrow 1, e_1 \leftarrow 0;$

$i \leftarrow 1$

AS-3: $q_{i+1} \rightarrow \text{select}(x, q_i, b_i, d_i, e_i);$

$a_{i+1} \leftarrow b_i,$

$b_{i+1} \leftarrow a_i + q_{i+1}b_i,$

$d_{i+1} \leftarrow e_i + q_{i+1}d_i,$

$e_{i+1} \leftarrow d_i,$

$p_{i+1} \leftarrow q_i p_i + p_{i-1},$

$q_{i+1} \leftarrow q_i q_i + q_{i-1};$

AS-4: After a sufficient number of iterations go to step AS-4

otherwise $i \leftarrow i + 1$, and go to step AS-3;

AS-5: $\sin^{-1}x \leftarrow p_{i+1}/q_{i+1};$

(Kishor Trivedi)

6. SWITCHING THEORY AND LOGICAL DESIGN

(Supported in part by the National Science Foundation under Grant Number U.S. NSF-GJ-503.)

T.K. Liu completed his Ph.D. thesis on synthesis of logic networks with MOS complex cells. Synthesis problems under different design criteria were discussed.

J.J. Mora-Touar started to work on the effect of additional inequalities on the computation time of logical design by integer programming, as his Master degree thesis. When logical design of optimal networks is done by the implicit enumeration method many additional inequalities derived from the intrinsic properties of networks were incorporated. He wanted to find what types of additional inequalities speed up the computation in different design problems.

Transformation techniques to improve given networks were developed by J.N. Culliney, Y. Kambayashi and H.C. Lai. Many interesting results were found.

Optimal adder networks under different conditions were designed by K. Hohulin. Also the manual of logical design of optimal networks by the implicit enumeration method based on the all-interconnection formulation was continued.

S. MUROGA

During this quarter, we completed programming and several algorithm refinements of our new NOR-gate network transformation procedure designated Phase II. The group of several somewhat simpler (in terms of algorithm complexity) network transformation procedures mentioned in previous Quarterly Progress Reports are referred to as Phase I procedures.

At this moment, Phase II seems, in general, much more powerful (in terms of the ability to reduce the cost of a network to realize a given function) than any single Phase I procedure, but our computational experience in comparing the two phases is still, admittedly, limited.

We have been experimenting with many different versions of Phase II and also with different methods of application of the procedure. We have also been considering combinations of Phase I procedures with Phase II to create a "system" of network transformation procedures with which to systematically attack a given network minimization problem.

Phase II offers many exciting new possibilities for network transformations not the least of which is the theoretical ability to transform a network which does not even realize the desired function (although it should realize a function somewhat "close" to the desired one) into a (hopefully) "near optimal" network which does realize the desired function.

(Jay Culliney)

The revision of the one-bit full adder paper which is to be published in IEEE TC has been completed. The revision of the first draft of the users manual for our all-interconnection formulation network synthesis program is nearly complete. Also, some time was spent working with J. Mora on the problem of determining the effect on the computation time of the additional inequalities used in our network synthesis programs.

(K. Hohulin)

Our main purpose is to make a logical design system which consists of the following two steps:

(1) Synthesis of a logic circuit to realize given function.

(2) Usage of network transformation procedures to obtain a lower cost network realizing the same functions.

This system is fit for designing a logic circuit which requires many gates, for example, a logic circuit for LSI (large scale integrated) circuit.

For step (2) we already have several procedures. During the last quarter period we developed a new algorithm for network transformation which uses an idea of error compensation, and the algorithm was tested on several networks by hand. By these examples the algorithm seems very effective. During this period the algorithm was modified in order to make it fit for computer programs, because our design system is not interactive. The algorithm for hand computation contains several criteria which are not readily adaptable in a computer program. Instead of simplifying these criteria, some new features were added. We have now several versions of this algorithm (the results of the program are discussed in the Quarterly Progress Reports of Mr. H. Lai and Mr. J. Culliney).

For step (1) we can use the following methods:

- (a) A three-level NOR network obtained from a two-level AND-OR network (procedures to synthesize such a network were discussed by many authors).
- (b) A three-level NOR network obtained by Gimpel's procedure.
- (c) Map factoring method.

Redundancies of the network obtained by step (1) may be removed by step (2), so for step (1) we can use simpler procedures which may produce a network with more redundancies than a network obtained by (a), (b) or

(c). For this purpose the following additional methods are considered:

- (d) The first solution of a branch and bound method.
- (e) Universal network.
- (f) A simplified procedure to synthesize a 3-level NOR network.

A procedure for (f) was developed during this period. This procedure will be used in a logical design system with a maximum level restriction.

For the synthesis of a network using ECL (emitter coupled logic), some modifications were added to a procedure using integer programming. By these modifications we can treat a larger network in the same amount of calculation time. Additional calculations for wired logic were also done during this period.

(Y. Kambayashi)

The main effort was put on developing a NOR-network transformation program by the so-called error-compensation approach. (See Quarterly Report by Y. Kambayashi, April-May-June, 1972). By this approach the NOR gates of a network may be removed one at a time and the resulting networks, which may no longer realize the desired function, are examined to see whether or not they can be transformed to realize the desired function by compensation of errors in each gate. A FORTRAN program for this procedure of compensation of errors is devised and coded. Although this program is not completely debugged yet, it seems to be quite powerful for transforming NOR networks into ones with smaller numbers of NOR gates.

(H.C. Lai)

Publications

Shinozaki, T., "Computer program for designing optimal networks with MOS gates", Department of Computer Science, University of Illinois, April, 1972, 118 pages, Master Thesis, Report No. UIUCDCS-R-72-502. (This discusses which parts of a design algorithm of MOS networks are most time-consuming, by implementing Liu's algorithm by computer programs. Then a revision of the algorithm is proposed.)

Liu, T.K., "Synthesis of logic networks with MOS complex cells", Department of Computer Science, University of Illinois, May, 1972, 227 pages, Ph.D. Thesis, Report No. UIUCDCS-R-72-517. (This discusses algorithms for synthesizing optimal two-level, multi-level feed-forward MOS networks. Also synthesis of multi-level feed-next MOS networks is discussed. Then an algorithm is developed to synthesize optimal feed-forward MOS networks with MOSFETS distributed among cells as uniformly as possible.)

Muroga, S. and T. Ibaraki, "Design of optimal switching networks by integer programming", IEEE TC, June, 1972, pp. 573-582.

7. OFFICE OF THE SOUPAC CONSULTANTS

(Statistically Oriented Users Programming and Consulting)

April - June 1972

A new chapter has been added to the history of the SOUPAC group. With the passing of the June Quarter, SOUPAC will be attached to the Computing Services Office. Future goals are already being discussed in this context. Much wider use of SOUPAC via distant remote terminals is surely just ahead.

In June Computing Services added one of its programmers, Beth Richardson, to the SOUPAC Staff. She is presently working on the Spectral Analysis Package. Three lectures were given during May to the Mathematics 461 class on Spectral Analysis. In addition, four new Spectral Analysis procedures will be released in the near future. They are as follows:

Period -- which calculates sample periodogram of a time series realization

FPeriod -- fast Fourier transform of above

Covariances -- calculates autocovariances and autocorrelations

Spectral -- calculates sample spectral density function of a time series realization

The SOUPAC System Programmer's Guide went to press as the final DCS report of the SOUPAC Group before joining CSO. This volume contains sections on the following:

The Executive Monitor
The Macro Library
The System Subroutine Library
The Syntax Interpreter

The Guide will probably not be in general distribution but will be useful mainly to future Systems Programmers and other installations receiving SOUPAC.

The counter in SOUPAC indicated an average of over six thousand jobs per month were run during the quarter, with nearly eight thousand run during May. This is an average of over 200 jobs per day.

Other projects of the SOUPAC group included continued dynamic allocation of programs, planning improvements to Express SOUPAC, improvements in the Transformations program, planning a Factor Scores program, planning a new Random Number Generator, developing a general regression hypothesis testing package, improvements to the Regression package, and release of Polynomial Regression program.

Retrospect and Prospect

Since joining the Department of Computer Science on February 1, 1968, the statistical programs of SOUPAC were rewritten to run on the IBM36 mod 75, and then they were revised again to run in variable region sizes. The SOUPAC Seminar has been offered almost every semester since the Spring of 1970. The SOUPAC manual has had five editions as a DCS report. More than twenty new programs were added to the library, and more than twenty different classes and groups have received materials and/or lectures from the SOUPAC Staff. (Note during the last academic year we have not distributed the manual ourselves, so we have no record of its distribution as to classes.)

The SOUPAC group is pleased to have been a part of the Department of Computer Science. Some of our most rapid development has taken place in the atmosphere of computer competence available here. Certainly the Group will continue serving the Department as well as the University community in their statistical programming and consulting needs. We look forward to continuing opportunities to serve the statistical needs of the computing public through the statewide network of CSO.

(Kern W. Dickman)

8. MACHINE AND SOFTWARE ORGANIZATION STUDIES

(Supported in part by the National Science Foundation under Grant Number US NSF GJ 27446)

The following is a collection of related work aimed at improved designs for computer and software systems. We are interested in parallel and pipeline processors, small primary memories, effective use of rotating memories, and some questions concerning user languages for problems including typical FORTRAN type calculations, simulation languages, and a variety of file processing problems.

(D. Kuck)

8.1. FORTRAN Parallelism Detection

8.1.1 The Analyzer - (S. C. Chen, J. Claggett, W. Hackmann, D. Lawrie, R. Strebendt, W. Tao, and R. Towle)

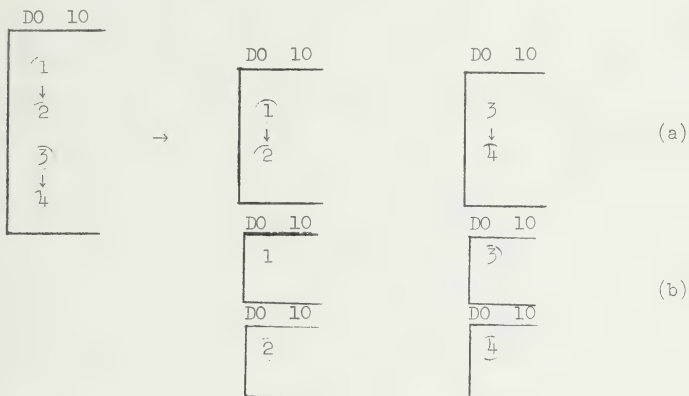
The original DO subprogram, incorporated with all other subprograms in analyzer, has been tested for many real FORTRAN programs and the result of that measurement has been collected during the last period.

During this period, a new strategy of cutting/splitting a DO loop in order to take advantage of forward substitution across iterations has been implemented. The algorithm is outlined as follows:

- (1) Find the dependence graph of the loop.
- (2) Separate this graph into complete disconnected subgraph and execute them in parallel.
- (3) If, within each subgraph, they are all assignment statements, then we perform forward substitution across iterations to exploit its optimal parallelism by using tree-height reduction techniques (in ASSIGN subprogram). If there are inner DO loops, then forward substitution and tree-height reduction is performed only on those blocks of assignment statements intervening these inner loops inside the same iteration. And then,

using the same strategy, we perform step (1), (2), and (3), until we exploit all parallelism inside each inner loop. After that, we can concatenate all inside assignment blocks and DO blocks to perform sequentially w.r.t. the outside DO loop.

The following simple example will show the difference between the old (b) and new (a) result:



This new implementation is being tested, and from the results we got so far it seems to give a better efficiency for big parallel machines, in comparison to the old strategy.

The SCANNER got some minor work done on it; specifically, correcting the DO nest count, constructing a table giving variable counts, operator counts, fixed and floating point counts, etc., for BAS.

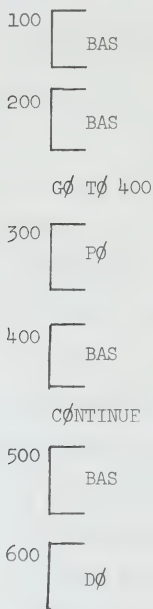
IFTREE seemed to have a bug, but analysis showed the problem to lie in ASSIGN--a subscript-range error. Minor changes were made to IFTREE.

Another statistics problem was solved in P50CALC. We wanted to know (trace-by-trace) T(1), P(MAX), T(MIN), S(P), etc., with respect to blocks of

similar types, i.e., how much time and how many processors were used in a trace for IFs, for DOs, for BAS? A table was constructed giving these statistics. A statistical bug was ironed out of LIKTYPS.

Several new capabilities have been added to the assignment block analysis routines.

1. Blocks of assignment statements which are adjacent, are separated by a "safe" reserved word (such as "~~C~~ONTINUE"), or are linked by an unconditional branch ("~~G~~O ~~T~~O") are collected and treated as a single block of assignment statements up to a size of thirty statements. For example, in the program segment



Blocks 100, 200, 400, and 500 would be combined to form block 100'.

Blocks 200, 400, and 500 would form block 200'. Blocks 400 and 500

would form block 400'. This effectively gives us the following

structure:

100'

BAS

GØ TØ 600

200'

BAS

GØ TØ 600

300

DØ

400'

BAS

GØ TØ 600

500

BAS

600

DØ

2. Under restricted conditions an assignment block which is between two other blocks (such as DØs) is moved to combine with an assignment block prior to the other blocks. For example,

BAS

DØ

BAS

DØ

is converted to

[BAS

[BAS

[DØ

[DØ

when it is possible to do so.

3. An extrapolation technique was added to the ASSIGN program to estimate the number of processors and the tree height for DO blocks whose number of iterations exceeds the capacity of the ASSIGN and TREEHT programs.

4. Numerous bugs have been fixed and code improvements have been made.

PTRACE (path tracing routine) has been reorganized into one main driver calling different routines to handle different statement types with a collection of supporting utility routines (mainly data collection and print out) instead of one big procedure doing everything. Ranges and averages of traces are also printed out now.

Also, whatever needs to be patched up and polished off has been done, especially in HISTØGM.

Then several routines have been added (the main one being DOPATH) to handle multiple-exit DO block record.

Also, a routine to check duplication of loops has been added.

A new HISTØgram routine is written (with according modification in ways of collecting data in path tracing) to print out statistics with both PE and TL as parameters.

A program was written to handle IFs inside DO loops. Also, the size of DO loops has been increased to 100 statements. More static information about FTH programs is coming from the scanner.

81.2 Parallelism Exploitation and Scheduling - (P. W. Kraska)

Parsing algorithms are developed such that syntactic tree-height is minimized, or reduced, with respect to application of the associative, commutative, and distributive (but not factoring) laws of arithmetic, on arithmetic expressions composed of well-formed sequences of the symbols add, subtract, multiply, divide, scalar identifier, and assignment, where it is assumed that a unique weight (i.e., program execution time) may be associated with each symbol. A parsing algorithm is also developed such that syntactic tree-height is minimized, with respect to application of the associative law, on expressions composed of a conformable sequence of matrix products, where the matrices are not necessarily square, and such that the overall number of computer operations is minimized if tree-height is not affected.

A new non-preemptive scheduling algorithm of a weighted-node acyclic dependency graph, having n nodes, on a system of k equipotent machines is developed such that all nodes are processed in the least amount of time. The assignment process requires $O(n^2)$ computer operations. A new algorithm to determine $k^* = \text{LUB}(k)$, such that the graph may be processed in the critical time, is also presented. Finally, it is shown how to optimally schedule a system of special purpose machines, where there are k_i equipotent machines of each type, in a graph.

2. Text Searching - (I. Chan, R. Goldberg, L. Hollaar, E. Polley, J. Rinewalt, W. Stellhorn, and R. Zears)

Until recently, the development of D-Machine software for input and output control and the subsequent testing of hardware interfaces has gone somewhat more slowly than was anticipated. Consequently, it has not been possible to test the text-handling routines as planned. However, several hardware malfunctions have been accounted for, and it is now possible to read cards reliably and display their

contents on the teletype. Control routines for teletype input and output have both been demonstrated, and a routine for line printer output is under development. With these facilities it will be possible to test S-level programs including the text-searching routines and certain auxiliary programs which have been written in connection with them.

Meanwhile, considerable progress has been made in planning text-searching experiments and in developing the associated requirements for system measurements. In addition, several promising sources for different types of digitalized text have been located.

A user's guide to the S-Language has been written and will be published as Department of Computer Science Report No. 534. Included in the report is a description of the implementation techniques used in the assembler. Version 3.0 of the assembler is now supported.

3. Theory of Computation - (P. Budnik)

We have been able to relate the hyperarithmetical sets to what might be called practical properties of Turing machines (TM).

An Intuitive Interpretation of the Hyperarithmetical Sets

To formalize this we make heavy use of the fact that the output of a TM may be interpreted as the godel number of another TM.

Let T^x be TM godel number x .

Let T_k^x be the k th output of T^x and undefined if there is no k th output. (Choose the godel numbering such that T_k^0 is undefined for all k).

Now for any hyperarithmetical set S , there is a WF T^V and another T^Z such that

$w \in S \equiv Q(v, T_w^Z)$ is true where

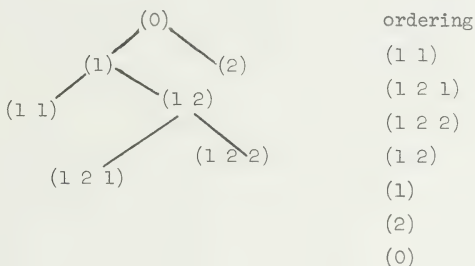
$$Q(x, y) = \text{Uk} (T_k^x = T_k^y = 0 \vee Q(T_k^x, T_k^y)).$$

$(\text{Uk } B(k) \equiv \forall k \exists m (B(n) \text{ is true for at least } k \text{ different integers } n \text{ all of which are less than } m)).$

These properties of TMs are practical in the sense that a compiler is a program whose output is interpreted as the godel number of a TM. A compiler compile is a TM whose outputs are compilers. The hyperarithmetic sets are natural generalizations of this notion.

This characterization allows us to provide a very concise formulation of Gentzen style consistency proofs. A principle of "nondeterministic recursive induction" can be stated. Consider the nondeterministic simulation of a TM and its outputs and their outputs, etc., until some specified termination symbol is output. Let each output be a pair of symbol strings, the first being a statement in arithmetic and the second either the godel number of a TM or a termination symbol. These outputs can be ordered by the index sets of the branches leading to the node in such a way that the branch closest to the root has highest weight but if one node is initial segment of the other the longest node is first.

Example:



Principle of Nondeterministic Recursive Induction

If a TM's nodes are such that each statement in arithmetic is either provable in the propositional calculus or of the form $\forall x A(x)$ where $A(0), A(1), \dots, A(k), \dots$ occur at predecessor nodes, then each statement at a node is true.

It is easy to construct a uniform algorithm which generates such a TM for any proof of a statement in arithmetic. Further, it is not difficult to see

how to extend arithmetic in terms of this principle without resorting to infinite sets.

These results were presented at the spring meeting of the Association for Symbolic Logic.

4. A Multiprocessor for Simulation Applications - (E. Davis)

Multiprocessor systems have generally been designed for applications with arrays of data which can be operated on in parallel. In this thesis an application area which does not contain such readily identifiable parallelism is examined. Discrete time simulation is found to contain several distinct levels at which potential for concurrent execution exists. The levels are used to guide the organization of a multiprocessor designed for simulation applications.

Both software and hardware aspects of the problem have been covered. Features of the system include a special processor used to evaluate conditional jump trees; clusters of simple, fixed point arithmetic processors; a unit to form and dispatch tasks to the processors; and a memory system which includes a read only program memory.

5. Elimination of Rotational Latency by Dynamic Disk Allocation - (D. E. Gold)

In Input/Output bound computer systems which use disks for back up memory, rotational latency is frequently the source of the I/O boundness. This paper presents several methods for eliminating rotational latency in head-per-track disks or equivalent memory devices. These methods assume pre-run time knowledge of the I/O sequences required by a particular program which will be run on the system. They work by constantly reorganizing the data on the disk during execution of the program. Measures are obtained of certain system parameters which indicate the requirements for use of these techniques and the various tradeoffs may be examined and quantified. These results are useful both for

esigning a new operating system and for removing latency from an already
xisting operating system.

6. Theses -

Edward W. Davis, Jr., "A Multiprocessor for Simulation Applications," (Ph.D. Thesis) University of Illinois at Urbana-Champaign, Department of Computer Science Report No. 527, June 1972.

David E. Gold, "Elimination of Rotational Latency By Dynamic Disk Allocation," (Ph.D. Thesis) University of Illinois at Urbana-Champaign, Department of Computer Science Report No. 522, May 1972.

Paul W. Kraska, "Parallelism Exploitation and Scheduling," (Ph.D. Thesis) University of Illinois at Urbana-Champaign, Department of Computer Science Report No. 518, June 1972.

Eugene J. Polley, Jr., "An Assembler for Efficient File Manipulation," (M.S. Thesis) University of Illinois at Urbana-Champaign, Department of Computer Science Report No. 534, Aug. 1972.

9. COMPUTER SYSTEMS ANALYSIS

(Supported in part by the National Science Foundation under Grant No. NSF GJ 28289)

The goal of this research is the development of analytical tools for system modeling and analysis of real time computer networks. Priority assignment and job dispatching rules for a geographically distributed computer network are being investigated.

Computer Network Modeling (J. Fitzgerald)

We have developed mathematical tools to define the probabilities of overloading any center in a network given that the input rate to a common queue for that center is an independent random variable. Input of jobs to each center is controlled by predetermined probability data stored in tabular form. We make a decision to inhibit, reduce or maintain the present input rate from this predetermined information. We have also developed an efficient controller by extending the tabular data to facilitate the dispatching of jobs between different centers in the network.

Center Throughput Analysis (W. Barr)

Having developed a measure of cost effectiveness for evaluating computer center throughput, we have determined that our measure is very indicative of the level of activity in computing centers and the turnaround time for jobs in different priority classes. We have formulated an algorithm which allows the user to specify a series of deadlines for a job and associate with each deadline a relative reward. We have also formulated, for each center, heuristic scheduling and priority assignment algorithms which guarantee that the deadlines of the highest priority jobs will be met, where

possible. The algorithm further indicates those deadlines which are in danger of being missed. We then use the measure of cost effectiveness to perform load-leveling between centers and, as a result, achieve economic viability within the network computer.

In the queueing theory we have made progress toward solving the problem of a single server which processes tasks from a queue with R priority classes in which tasks are allowed to renege. This is the system which develops from our priority assignment and scheduling algorithms.

Publications

E. K. Bowdon, Sr., "Design Automation in Networks Computers," Ninth Annual Design Automation Workshop Proceedings, June, 1972, pp. 350-356.

Abstract:

Network computers offer the designer the potential of combining the advantages of resource sharing, data base sharing, and message switching with those of design automation. In this paper we present a brief description of a network computer currently under development and then propose an interactive design automation system for this geographically distributed network.

The control mechanism for our design automation system is provided by an operating system, resident in each computer, whose primary function is to segment, disseminate, and regulate the jobs generated by the design automation system. When a job enters the system it may be assigned to anyone of the computers in a center. The job then runs to completion under control of that computer, but the segments that make up the job may be executed in any available computer. Additionally, we may transmit jobs between centers to improve the network response time. Thus we see that by detecting when segments are ready to be executed and by queueing this available work uniformly among all computers in the network, our design automation system will give each designer access to the full parallel processing power of the network.

Fred Salz, "A GPSS Simulation of the 360/75 Under HASP and O.S. 360," Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, Report No. UIUCDCS-R-72-528, June 1972.

(E. K. Bowdon)

10. NUMERICAL ANALYSIS

It is apparent that positive definite factorization methods may be derived for irregular finite element matrices. This makes it possible to apply an iterative method due to M. Diamond* to these matrices. Whether an algorithm can be devised to do this for a class of irregular matrices is an important problem to study if factorization procedures may be applied to general finite element matrices. Perhaps equally important is the data management problem of manipulating sparse matrices in an iterative procedure. Code is being written for a specific example to implement this factorization procedure with Diamond's algorithm. The example is the flow problem that results in modeling a liquid waste well injection system.

(Paul E. Saylor)

*M. A. Diamond, An economical algorithm for the solution of finite difference equations, doctoral thesis, Department of Computer Science, University of Illinois, 1972, issued as report no. UIUCDCS-R-71-492, DCS, 1971.

11. COMPUTER LANGUAGES FOR MATHEMATICS AND NUMERICAL ANALYSIS
(Supported by the National Science Foundation under Grant No. US NEF
GJ-328).

The OL/2 array language is available for experimental use on the IBM 360. A preliminary user's manual is being prepared and will incorporate the latest additions to the language. A new input-output module has been added, allowing various types of arrays and subarrays to be displayed naturally. A number of array algorithms have been written and are available as examples of the OL/2 language. Assistance in the use or application of OL/2 can be obtained by contacting J. R. Phillips, 285 DCL.

SIGNIFICANT DIGIT ARITHMETIC

The implementation of basic software for significant digit arithmetic is contained in a report (UIUCDCS-R-72-530) by S. Lai. The implementation is based on the unnormalized arithmetic of Metropolis and Ashenhurst and incorporates the ABC arithmetic (Analyzed Binary Arithmetic) of Metropolis. The latter is a refinement of significant digit arithmetic and has the distinct advantage of separating precise from imprecise numbers. Algorithms using significant digit arithmetic in place of normalized arithmetic are very illuminating. There are many interesting features that are discussed in the literature as referenced in the above report. In the future the significant digit arithmetic will be an option in the OL/2 language. In the meantime, the basic routines are available for experimentation from the Ol/2 implementation group in Room 94, DCL. (J. Richard Phillips, principal investigator, R. Bloemer, D. Jurich, E. Tangman, and S. Lai.)

REFERENCES

Lai, Steven S. "Implementation of Basic Software for Significant Digit Arithmetic," Department of Computer Science, University of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R-72-530, June, 1972.

12. COMPUTATIONAL ASPECTS OF COMBINATORIAL ALGORITHMS

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 31222).

The most important aspect of our research on trees has been the development of a new class of binary search trees¹. These trees, which we call trees of bounded balance or weight-balanced trees are important because they are easy to maintain in their form despite the insertion and deletion of nodes; moreover, the search time is only moderately longer than in completely balanced trees. Trees of bounded balance differ from other classes of binary search trees in that they contain a parameter which can be varied so the comprise between short search time and infrequent restructuring can be chosen arbitrarily. We now intend to bring this part of our work to a conclusion, and we are preparing two survey papers which summarize the knowledge gained in this research. We have started an investigation of directed acyclic graphs as data structures; these can be viewed as a generalization of trees when the trees are allowed to have common job trees. We intend to continue this line of research during the next year.

A second facet of our research has been in the area of establishing the optimality of various algorithms. For example, we have discovered new proofs which establish lower bounds on the number of arithmetic operations required for polynomial evaluation². The previous proofs of these results are difficult to follow, or they involve the use of fairly deep results from topology. We have developed relatively simple, self contained proofs, for these theorems. In continuing our research on optimal algorithms, we

are currently studying the problem of determining whether a set of points in the plane form a convex polygon. Related are the problems ordering the points around the perimeter of the polygon and, if the points do not form a convex set, then determining their convex hull.

(Jurg Nievergelt/Edward M. Reingold)

References:

1. Nievergelt, J. and Reingold, E. M. "Binary search trees of bounded balance", Proc. Fourth Annual ACM Symp. on Theory of Computing, 1972.
2. Reingold, E. M. and Stocks, A. I. "Simple proofs of lower bounds for polynomial evaluation", Proc. Symp. on Complexity of Computer Computations, Plenum Publishing Co., 1972.

13. NUMERICAL METHODS, COMPUTER ARITHMETIC AND ARTIFICIAL LANGUAGES

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 812.)

13.1 GIZMO

Work has proceeded on the development of GIZMO, the CAI system for the PDP-11 under Educational Timesharing System (ETS). The teacher mode now permits the teacher to define lessons in BACKUS-NANR form and is permitted to nest definitions to an arbitrary depth as well as employ a definition recursively. The teacher may also specify text to be printed to the student if he types either "HELP" or "GIVE-UP".

The lesson compiler which accepts as input the product of teacher mode and delivers as output a lesson which is usable for student mode with a minimum of effort on student mode's part has been coded and debugged. Teacher mode, the compiler, and student mode are all fully operational now and are now undergoing expansion rather than modification.

The University public are invited to try writing lessons and to use them under student mode. Writing lessons is easy (see Teachers Manual for GIZMO which appeared in the First Quarter, 1972 Quarterly Progress Report) and can be learned in just a few minutes.

(Al Davis, Bill Holt)

13.2 Design Criteria for a High Level Language for System Programming

The increasing availability of cheap, fast, small computers in the past decade has made possible the use of such computers as special-purpose machines, often with a hard-wired program. Customarily, such mini-computers have been programmed at the assembly-language level--although for two machines to share an order code is a rarity. Such an

approach, involving cheap hardware and high programming costs, tends to negate the economic advantages of mini-computers. In a university research situation, one of the main requirements of an operating system is, in fact, its modifiability and flexibility. In this environment, the cost of re-programming can exceed the original cost of programming. These considerations indicate the desirability of a high-level, systems programming language, designed for writing operating systems in. This approach has been used, on medium-scale machines such as the B5500, B6500, and on the PDP-11 using PEESPOL, which is a B6500 resident cross-compiler for the PDP-11.

Operating systems as such tend to be concerned with tasks such as scheduling, device-level I/O, input-output formatting, filing system maintenance, and, usually, manufacturer supplied software. A language to handle these should have the features:

1. It must be algorithmic--it must be possible to express procedures clearly and precisely.
2. Programs written in it should tend to be self-documenting.
3. It must be possible, in some way, to explicitly manipulate addresses--on the other hand, it should be difficult to do so accidentally.
4. It must be possible to write re-entrant, "pure" code--since otherwise a multi-programming system cannot easily be written in it.
5. Very explicit control of I/O should be permissible--it should be feasible to write a file structure in it.
6. Character manipulation facilities should be provided--to allow for "JCL", as well as to permit compilers and assemblers to be written in it.

7. It should be transportable--implementations on other machines should be an easy extension. Ideally, it should have a compiler written in itself.
8. It should produce economical code--as distinct from optimized code. That is, it should allow a simple procedure entry implementation, efficient evaluation of arithmetic expressions, and efficient loop-control.
9. Complex data structures should be readily, and conveniently expressible.
10. Features such as subscript checking, and other debugging aids, should be available as an option.
11. If at all possible, it should be based heavily on an existent language--thus allowing interchange of program segments with other establishments.
12. Since it is desired to write a compiler of it on the PDP-11, there should be some evidence that a small compiler of it is possible.
13. Multi-tasking facilities should be available.
14. The language should be easily extensible.
15. Explicit control of storage allocation should be available.

Of these, PL/I, OSL/2, and PEESPOL are "large" languages, which might need considerable restrictions before a PDP-11 resident compiler becomes feasible, PL/C is a subset of PL/I excluding features, such as multi-tasking, record I/O and based storage, which are highly desirable, and ALGOL 60 has no I/O facilities.

B6500 ALGOL and PASCAL have, between them, every feature deemed desirable--both have compilers written in themselves, both can be made to produce good code on the PDP-11, and both can be made to do "record

I/O". The most suitable language, satisfying the above requirements, could therefore be described as B6500 ALGOL with PASCAL additions, or PASCAL with B6500 additions. Since most of the desirable features of B6500 ALGOL are direct add-ons to ALGOL 60, and PASCAL, while heavily based on ALGOL 60, has considerably improved declaration and I/O facilities, it was decided to implement an extended subset of PASCAL. (Since system programming applications tend not to use floating-point, and the PDP-11 has no standard f.p unit, real variables are being omitted for the first stage of implementation.)

CC6000 series PASCAL has now been brought up on the CERL 6400 and work is underway to transport this compiler to the PDP-11. A very simple multi-user operating system is being designed for the PDP-11 for this purpose.

(I. Stocks, R. Cattell, P. Emrath)

13.3 MIPS

The system is called the Multi-task Interpretive Paging System or MIPS to reflect the fact that user programs run under an interpreter which pages the user code in order to implement a virtual machine. MIPS was designed to serve a community of inexperienced student programmers taking an introductory course in assembly language programming. This application requires fast turnaround for small student jobs without precluding simultaneous use by system programmers and research personnel who may want to run jobs of several hour duration.

MIPS is intended for use on an existing target machine with a particular hardware configuration. The system includes a PDP-11/20 processor with 20K words of core memory, a 256K word fixed head per

track disk, a dual DECTape transport, several terminals including a console teletype, a 600 cpm card reader, and a 500 lpm line printer. Operating system development is somewhat complicated by the fact that the PDP-11/20 does not provide any form of memory protection, segmentation, or relocation hardware. MIPS is a batch system because, in the opinion of the author, this configuration is not well suited to time-sharing due to inadequate disk capacity. The monitor therefore does not attempt to support remote access from terminals.

In the MIPS multi-task environment, system tasks handle spooling of cards to disk and listings to the printer while user tasks run concurrently in two fixed partitions. To prevent long jobs from tying up the CPU, user tasks running in the larger of the two partitions are swapped out to disk after a preset time interval and are given additional time slices in round robin competition with other jobs in the system. MIPS supports a job mix of up to six jobs and manages all resources required for execution of these jobs. Disk management routines attempt to optimize disk access time to improve system performance. In general, fairly conventional multiprogramming techniques are used throughout the system design. All monitor code was written by the author and consists of some 4300 cards. It is a modular system to permit simple restructuring of partitions. To promote understanding, monitor code is extensively commented and avoids "clever" coding practices.

This system is fully documented in The Design and Implementation of a Multi-Task Batch Operated System and Paging Interpreter for the PDP-11 (87 pp), an M.A. thesis.

(J. D. Miller, Earl Heffley)

GIZMO

Work has proceeded on the development of GIZMO, the CAI system for the PDP-11 under Educational Timesharing System (ETS). The teacher mode now permits the teacher to define lessons in BACKUS-NANR form and is permitted to nest definitions to an arbitrary depth as well as employ a definition recursively. The teacher may also specify text to be printed to the student if he types either "HELP" or "GIVE-UP".

The lesson compiler which accepts as input the product of teacher mode and delivers as output a lesson which is usable for student mode with a minimum of effort on student mode's part has been coded and debugged. Teacher mode, the compiler, and student mode are all fully operational now and are now undergoing expansion rather than modification.

The University public are invited to try writing lessons and to use them under student mode. Writing lessons is easy (see attachment, Teachers Manual for GIZMO) and can be learned in just a few minutes.

(Al Davis, Bill Holt)

14. GENERAL DEPARTMENT INFORMATION

14.1 Personnel

The number of people associated with the Department in various capacities is given in the following table:

	<u>Full-time</u>	<u>Part-time</u>	<u>FTE</u>
Faculty	22	4	23.97
Visiting Faculty	3	0	3.00
Research Associates and Instructors	2	2	2.7 ¹ / ₃
Graduate Research Assistants	0	66	33.08
Graduate Teaching Assistants	0	28	14.575
Professional Personnel	9	1	9.50
Administrative and Clerical	19	0	19.00
Nonacademic Personnel (Monthly)	18	1	18.50
Nonacademic Personnel (Hourly)	<u>0</u>	<u>56</u>	<u>14.39</u>
TOTAL.....	73	158	138.758

The Department Advisory Committee consists of Professor J. N. Snyder, Head of Department, Professors E. K. Bowdon, D. F. Cudia, K. W. Dickman, M. F. Faiman, H. G. Friedman, C. W. Gear, D. B. Gillies, W. J. Kubitz, D. J. Kuck, B. H. McCormick, R. G. Montanelli, S. Muroga, T. A. Murrell, J. Nievergelt, J. R. Phillips, W. J. Poppelbaum, S. R. Ray, E. M. Reingold, J. E. Robertson, P. E. Saylor, D. L. Slotnick, J. E. Vander Mey, D. S. Watanabe, and T. Wilcox.

During the second quarter, the following publications were issued by the laboratory:

Report Numbers

No.	523	Oxley, Donald Wayne, "ETS System User's Guide," May, 1972.
No.	524	Tse, Bernard, "Supplementary Information on <u>VISTA</u> ," June, 1972.
No.	528	Salz, Fred, "A GPSS Simulation of the 360/75 Under HASP and O.S. 360," June, 1972.
No.	529	Chouinard, Paul and with Forward by Dickman, Kern W., "SOUPAC System Programmer's Guide," June 28, 1972.

Theses

No.	497	Read, John S., "Parallel Image-Processing for Automated Cervical Smear Analysis," June, 1972 (M.S. Thesis).
No.	502	Shinozaki, Teruaki, "Computer Program for Designing Optimal Networks with MOS Gates," April, 1972 (M.S. Thesis).
No.	508	Cunningham, Ian MacDonald, "Hardware/Software Interface for the Stereomatrix Display," June, 1972 (M.S. Thesis).
No.	511	Wallman, Lawrence Henry, "BLAST: A Stereoscopic Television Display," June, 1972 (Ph.D. Thesis).
No.	512	Hanson, Donald Farness, "Outlining and Shading Generation for a Color Television Display," June, 1972 (M.S. Thesis).
No.	513	Sanford, Lynne Schaber, "Array Pattern Matching," June, 1972 (M.S. Thesis).
No.	514	Schwebel, John C., "A Graph-Structure Transformation Model for Picture Parsing," May, 1972 (Ph.D. Thesis).
No.	515	Kato, Takehiko, "Raser: RAndom to SERial Converter," June, 1972 (Ph.D. Thesis).
No.	516	Bracha, Nurit, "Transformations on Loop-Free Program Schemata," June, 1972 (Ph.D. Thesis).
No.	517	Liu, Tso-kai, "Synthesis of Logic Networks with MOS Complex Cells," May, 1972 (Ph.D. Thesis).

Theses Cont'd

No.	518	Kraska, Paul W., "Parallelism Exploitation and Scheduling," June, 1972 (Ph.D. Thesis).
No.	519	Wang, Benjamin Shaw-hu, "Monte Carlo Simulation of Non-linear Radiation Induced Plasmas," May, 1972 (Ph.D. Thesis).
No.	520	Oxley, Donald Wayne, "The Design and Implementation of an Educational Timesharing System for the PDP-11," June, 1972 (M.S. Thesis).
No.	522	Gold, David E., "Elimination of Rotational Latency by Dynamic Disk Allocation," May, 1972 (Ph.D. Thesis).
No.	525	Trivedi, Kishor Shridharbhai, "An Algorithm for the Solution of a Quadratic Equation Using Continued Fractions," June, 1972 (M.S. Thesis).
No.	527	Davis, Edward Willmore, Jr., "A Multiprocessor for Simulation Applications," June, 1972 (Ph.D. Thesis).
No.	530	Lai, Steven See Sun, "Implementation of Basic Software for Significant Digit Arithmetic," June, 1972 (M.S. Thesis).

File Number

No.	870	van Melle, Bill, "An Operating Manual for DIFMF3," April, 1972.
No.	871	Read, John S., "Image 8: Microscopic Stage and Focus Motor Control," June 30, 1972.

14.3 Colloquia

"Techniques of Design Automation," by Dr. Sheldon E. Akers, Electronic Systems Laboratory, General Electric Company, Syracuse, New York, April 10, 1972.

"Conditional Bit Sampling," by Professor John S. Sobolewski, Departments of Electrical Engineering and Computer Science, Washington State University, Pullman, Washington, May 4, 1972.

"Recent Work at the Stanford Hand-Eye Project," by Professor Jerome A. Feldman, Computer Science Department, Stanford University, Stanford, California, May 8, 1972.

"Approximate Solution of Dirichlet's Problem Using Approximating Polygonal Domains," by Professor Vidar Thomée, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin, May 15, 1972.

"On the Design of Multiple-Fault Diagnosable and Self-Checking Logic Networks," by Professor Gernot Metze, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, May 22, 1972.

14.4 Drafting

During the second quarter, a total of 337 drawings were processed by the general departmental drafting section:

Formal Drawings

Large Drawings	102
Medium Drawings	42
Small Drawings	85
Layouts	0
Report Drawings	78
Change Order Drawings	8
Miscellaneous Drawings	<u>22</u>
Completed Total Drawings.....	337

(M. Goebel)

14.5 Shop's Production

Job orders processed and completed during the second quarter of 1972 are as follows:

	<u>AEC 2118</u>	<u>AEC 1469</u>	<u>Other</u>
Machine Shop	5	11	2
Electronic Shop	1	144	32
Chemical Shop	1	110	5
Layout Shop	1	71	1

(F. P. Serio)

DEPARTMENT OF COMPUTER SCIENCE
GRADUATE COLLEGE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

REPORT REQUEST FORM

Report Number

Title

Fold and Staple as shown

NAME:

ADDRESS:

Fill out Mailing Label



NAME:

ADDRESS:

CUT ALONG THIS LINE

FOLD

Mail Room
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

FOLD

STAPLE

BIBLIOGRAPHIC DATA 1. Report No. UIUCDCS-QPR-72-2		2.	3. Recipient's Accession No.
Title and Subtitle Quarterly Progress Report		5. Report Date	
		6.	
Author(s)		8. Performing Organization Rept. No.	
Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801		10. Project/Task/Work Unit No.	
		11. Contract/Grant No.	
Sponsoring Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801		13. Type of Report & Period Covered Quarterly Progress Report - Apr.-June '72	
		14.	
Supplementary Notes			
Abstracts An abstract is not applicable.			
Key Words and Document Analysis. 17a. Descriptors Not Applicable			
Identifiers/Open-Ended Terms Not Applicable			
OSATI Field/Group			
Availability Statement RELEASE UNLIMITED		19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 223
		20. Security Class (This Page) UNCLASSIFIED	22. Price



the 1990s, the number of publications on the topic of the effects of the environment on human health has increased rapidly. The number of publications in this field has increased from 10 in 1980 to 100 in 1990, and to 150 in 1995.

One of the main reasons for this increase is the growing awareness of the importance of the environment for human health. The World Health Organization (WHO) has estimated that the environment is responsible for about 25% of the global burden of disease. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease.

The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease. The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States.

The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease. The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States.

The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease. The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States.

The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease. The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States.

The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease. The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States.

The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease. The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States.

The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States. This estimate is based on a study of the burden of disease in the United States, which found that the environment was responsible for about 25% of the total burden of disease. The WHO study found that the environment was responsible for about 25% of the total burden of disease in the United States.

Physics

510.84

Il6t

Real Closet

QUARTERLY TECHNICAL PROGRESS REPORT

July, August, September 1972

THE LIBRARY OF THE
FEB 22 1973
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

THE LIBRARY OF THE
FEB 22 1973
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

QUARTERLY TECHNICAL PROGRESS REPORT

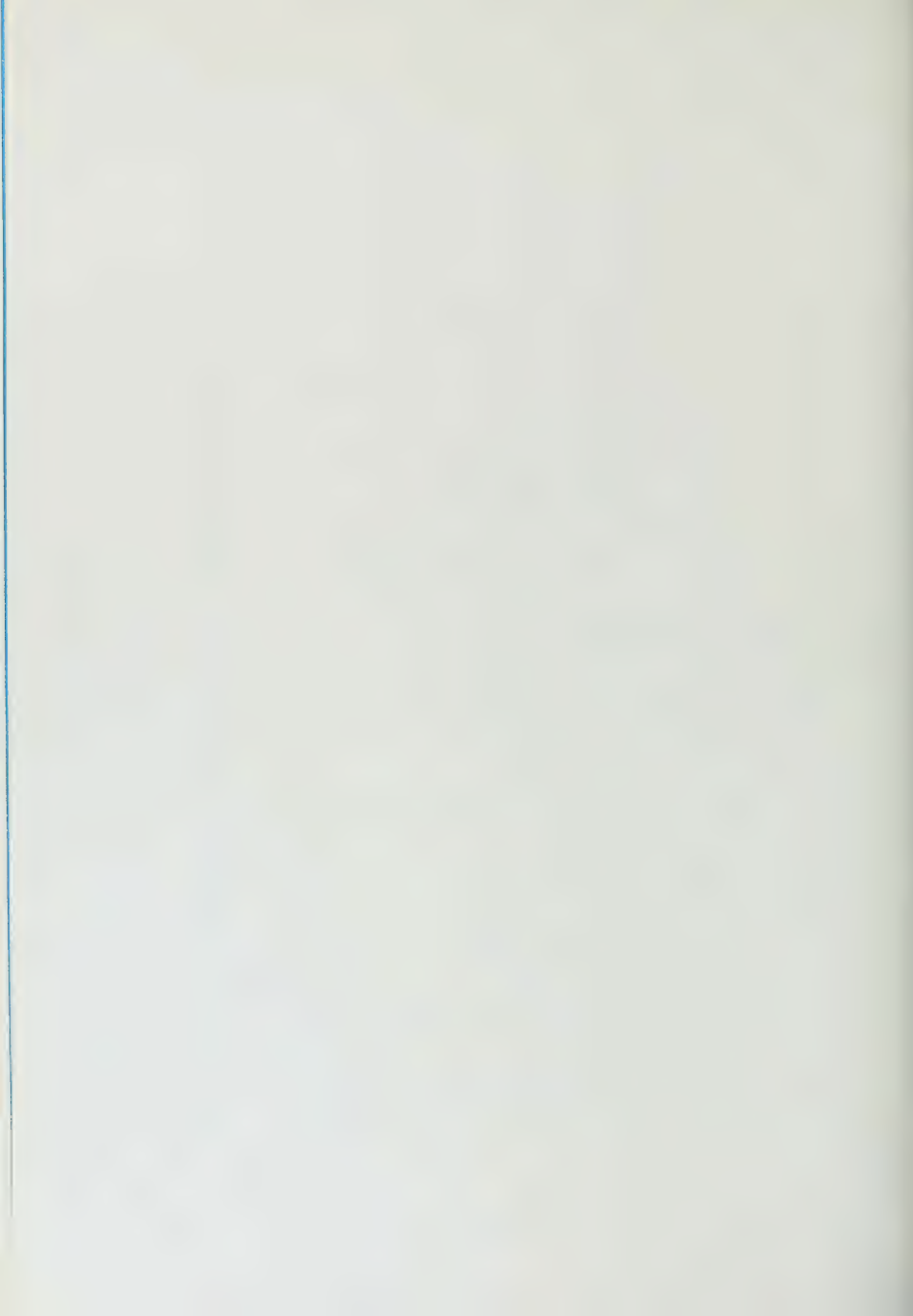
July, August, September

UIUCDCS-QPR-72-3

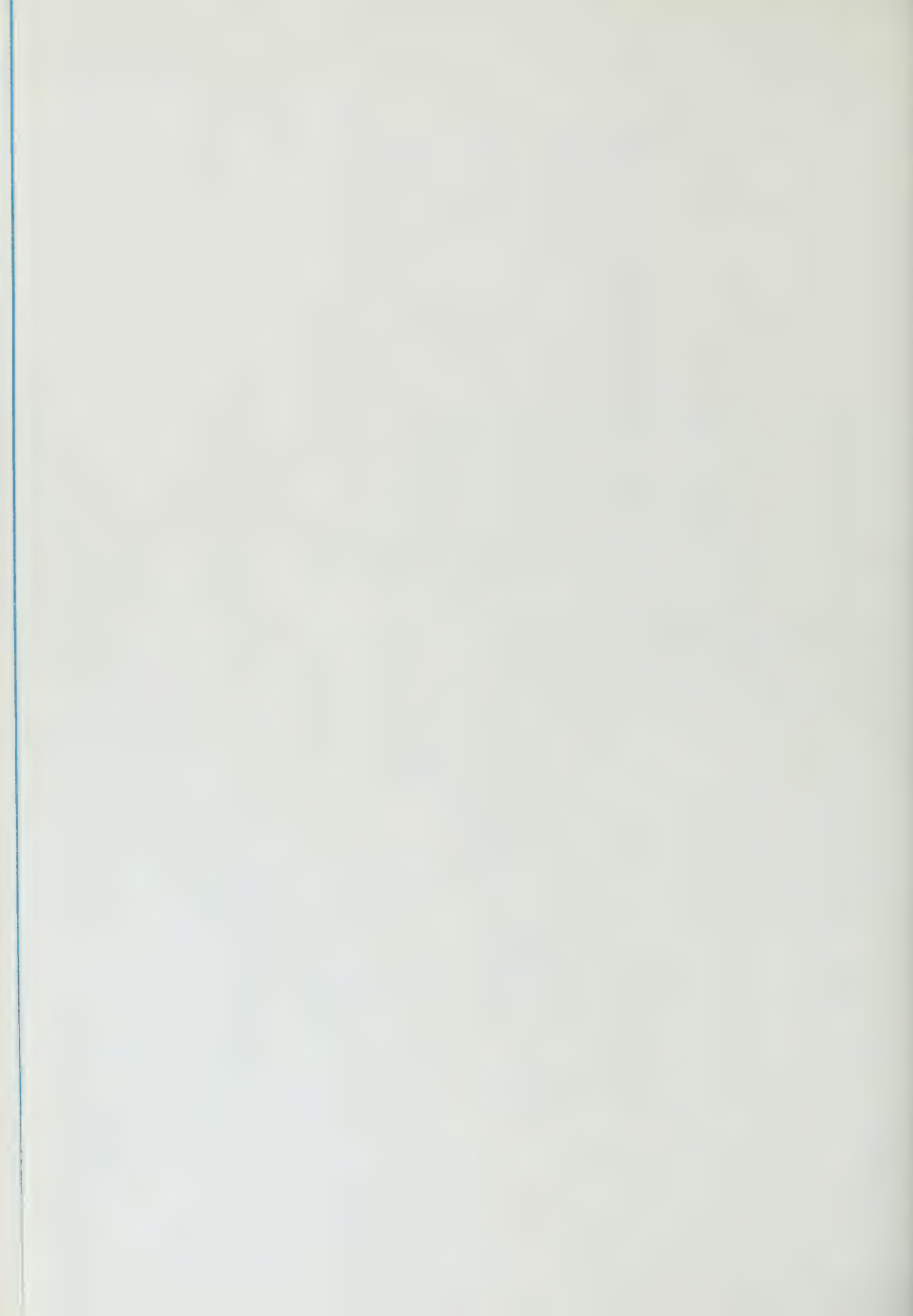
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

TABLE OF CONTENTS

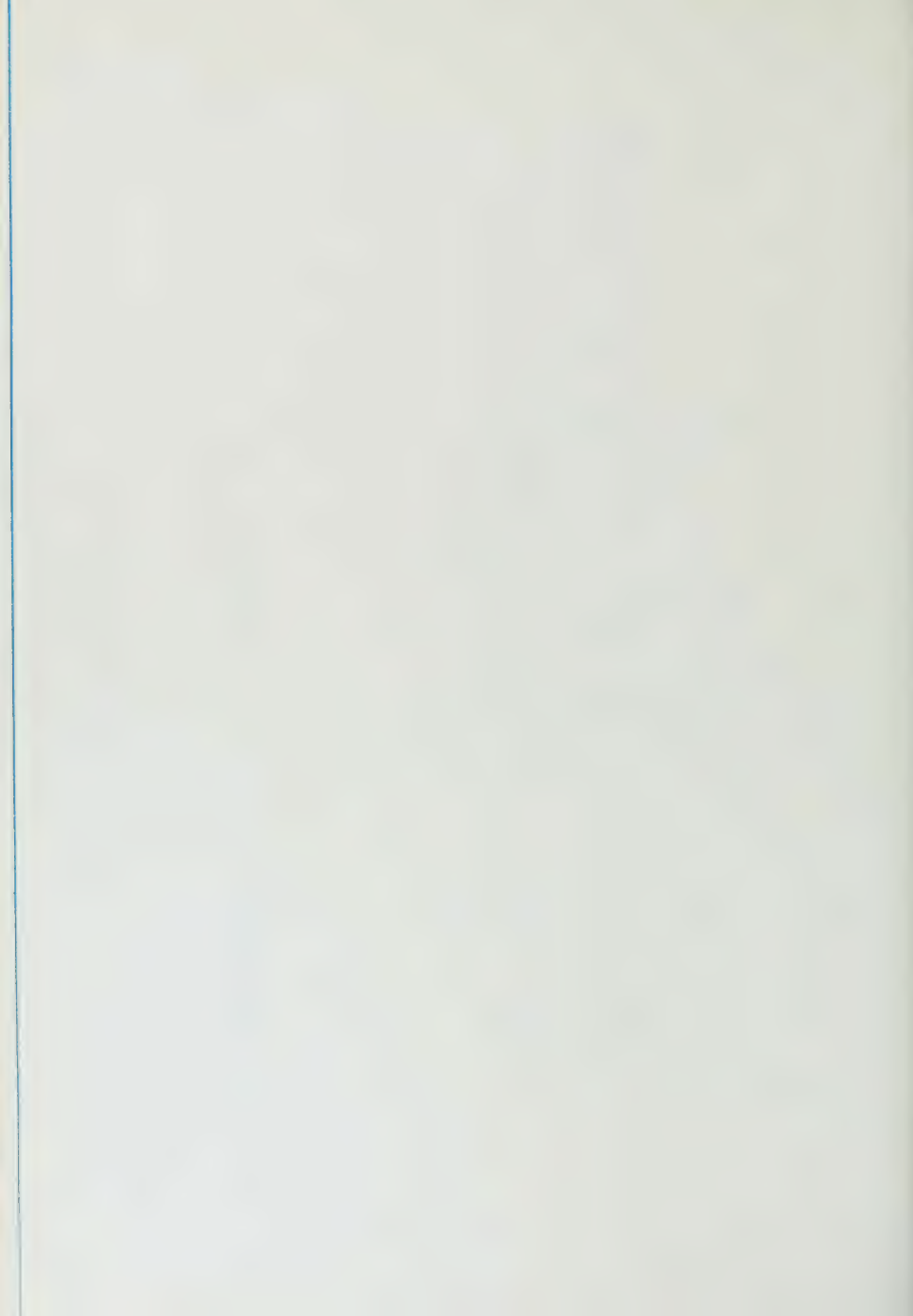
	Page
1. CIRCUIT RESEARCH.	1
1.1 APE (Project No. 25).	2
1.1.1 The Remote Power Supply	1
1.2 PENTECOST (Project No. 31).	3
1.2.1 Summary	3
1.2.2 Vertical Size Control	5
1.2.3 Video Analog Switch	7
1.2.4 Horizontal Deflection Control	7
1.3 Ergodic (Project No. 39).	10
1.3.1 Summary	10
1.3.2 Arithmetic Unit	10
1.4 Telemaze (Project No. 41).	15
1.4.1 Project Summary	15
1.4.2 Major State Generator	15
1.4.3 Video Gray Scale Generator.	15
1.4.4 Communication System.	17
1.5 INCOM (Project No. 46).	20
1.5.1 Introduction.	20
1.5.2 A Nodal Analysis Technique.	22
1.5.3 Dealing with Nonuniformity.	24
2. HARDWARE SYSTEMS RESEARCH	26
2.1 IASCO (Project No. 09)	27
2.2 OLFT (Project No. 12)	27
2.3 Sematrix (Project No. 24).	28
2.4 LINDA (Project No. 28).	28
2.4.1 Summary	28
2.4.2 Project Status.	28
2.4.3 Future Work	30
2.5 Stereomatrix (Project No. 30)	30
2.5.1 Changes to Coefficient Generator.	30
2.5.2 Display	32
2.6 Scantrix (Project No. 35)	32
2.6.1 Enabling the Display.	32
2.7 FROG (Project No. 36)	36
2.7.1 Simulation Study.	36
2.7.2 Hardware Implementation	36
2.8 REACON (Project No. 38)	36
2.8.1 Amplifying Cell	36
2.8.2 Sequential Freeze Circuit	37
2.9 OCOMO (Project No. 42).	37
2.9.1 Summary	37
2.9.2 Present Work.	40
2.10 Caecotron: A Tube for the Blind (Project No. 43)	44



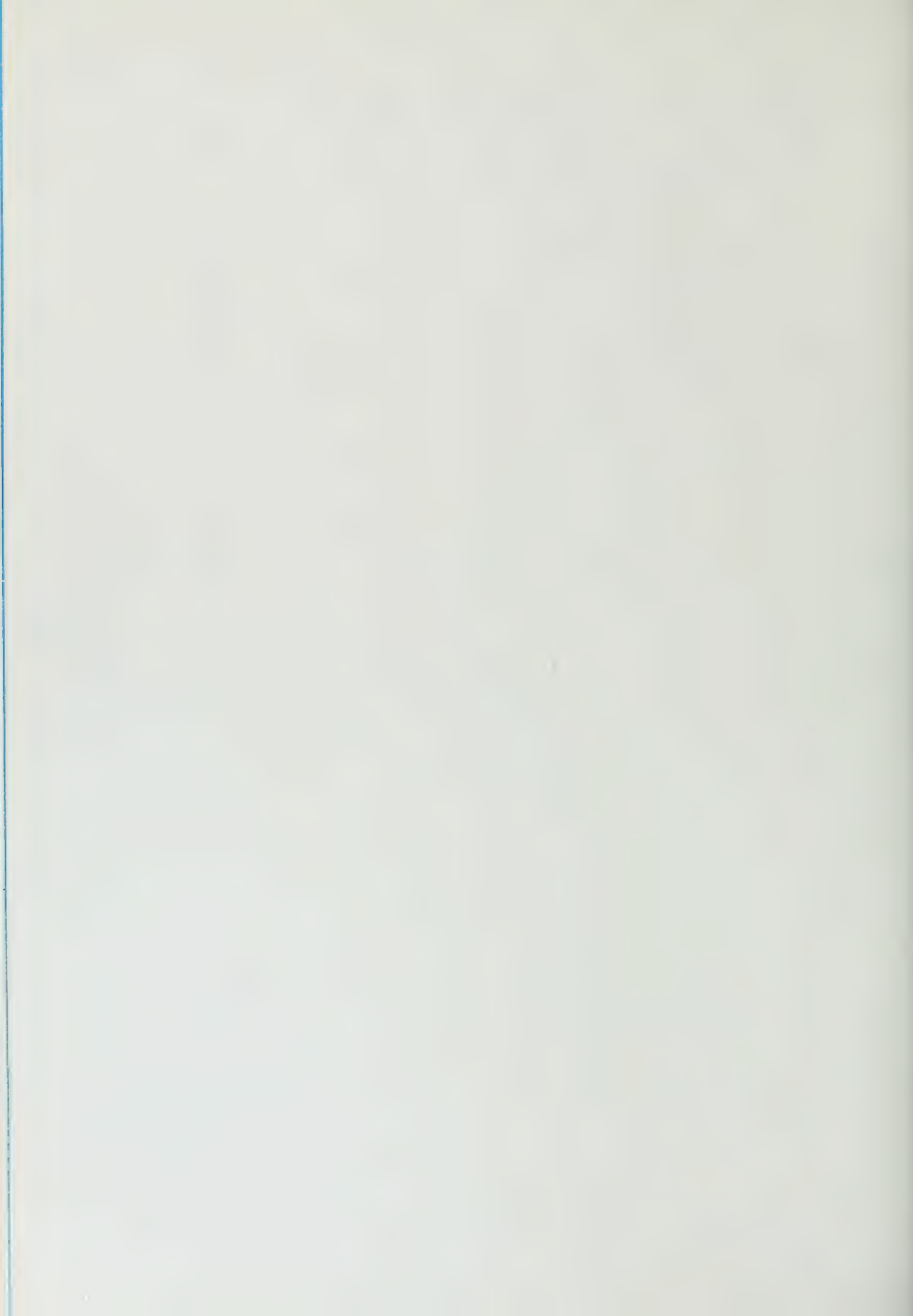
	Page
3. SOFTWARE SYSTEMS RESEARCH.	48
3.1 Numerical Processes.	48
3.1.1 DIFSUB	48
3.1.2 Sparse Eigen Problem	52
3.1.3 Auto-Restart	54
3.1.4 Plot Package	55
3.1.5 Item Analysis.	57
3.1.6 FINDEX	58
3.2 Illinois Graphics Computing System (IGCS). .	74
3.2.1 Overlay Supervisor	74
3.2.2 A Drafting Language (ADL).	74
3.2.3 Plot Sectorization Overlay	78
3.2.4 ILLISM E	79
3.3 Graphical Remote Access Support System . . .	79
3.3.1 OS/8 Operating System.	79
3.3.2 Information Retrieval.	80
3.3.3 Monitors	80
3.3.4 GLOM: New Filing System	82
3.4 Computer Maintenance and Construction. . . .	88
3.4.1 Graphics-8 Hardware.	88
3.4.2 Equipment Maintenance Log Summary. . . .	89
3.4.3 PDP-7 and 630 Maintenance.	89
4. CENTER FOR ADVANCED COMPUTATION.	92
4.1 Algorithm Development Group.	93
4.1.1 Introduction	93
4.1.2 Computational Methods in Linear Algebra.	93
4.1.2.1 Solution of Systems of Linear Equations . . .	93
4.1.2.2 The Algebraic Eigenvalue Problem.	93
4.1.3 Linear Programming	94
4.1.4 Graph Algorithms	95
4.1.5 Approximation of Functions	95
4.1.6 Finding Real Roots of Polynomials.	96
4.2 ILLIAC IV Multispectral Image Processing . .	97
4.2.1 Introduction	97
4.2.2 Research Findings.	97
4.2.3 Proposed ILLIAC IV Implementation.	98
4.3 ILLIAC IV Language Development for the Phase II System.	99
4.3.1 Introduction	99
4.3.2 IDOL	99
4.3.3 Types of Problems.	100
4.4 Economic Research Group.	102



		Page
	4.4.1 STEP I	102
	4.4.2 STEP II.	102
	4.4.3 MEASURE.	102
..5	Network Systems Group.	104
	4.5.1 Introduction	104
	4.5.2 Project to Interface the B6700 to the ARPA Network	104
	4.5.3 ARPA Network Activities.	105
	4.5.3.1 ARPA Network Terminal System (ANTS) Develop- ment	105
	4.5.3.2 ARPA Network Usage . .	106
	4.5.3.3 Further Installations of ANTS Systems on the Network.	106
	4.5.4 Graphics Support for Center Projects on the Burrough B6700.	107
	4.5.5 Network Graphics Efforts	107
	4.5.5.1 Network Graphics Protocol	107
	4.5.5.2 Laboratory for Atmo- spheric Research Support.	108
4.6	Administration	109
	4.6.1 Introduction	109
	4.6.2 Fiscal Status.	109
5.	THEORY OF DIGITAL COMPUTER ARITHMETIC.	113
	5.1 Study of Logical Organization for LSI Implementable Arithmetic Units	113
	5.2 Automatic Evaluation of Some Elementary Functions Using Microprogramming	113
	5.3 Implementation of Continued Product Algorithms	114
6.	SWITCHING THEORY AND LOGICAL DESIGN.	115
7.	MACHINE AND SOFTWARE ORGANIZATION STUDIES.	120
	7.1 FORTRAN Parallelism Detection.	120
	7.2 Theoretical Bounds on Parallism.	123
	7.3 Switching Networks	124
8.	COMPUTER SYSTEMS ANALYSIS.	125
	8.1 Computer Network Modeling.	125
	8.2 Center Throughput Analysis	125
9.	NUMERICAL ANALYSIS	128



	Page
10. COMPUTER LANGUAGES FOR MATHEMATICS AND NUMERICAL ANALYSIS	131
10.1 Array Expressions in OL/2.	131
10.2 Compiling and Executing OL/2 Programs.	132
10.3 Examples of OL/2	132
11. GENERAL DEPARTMENT INFORMATION	139
11.1 Personnel.	139
11.2 Bibliography	140
11.3 Abstracts.	143
11.4 Colloquia.	152
11.5 Drafting	153
11.6 Shop's Production.	153



1. CIRCUIT RESEARCH

(Supported in part by the Office of Naval Research under Contract
N000 14-67-A-0305-0007, W. J. Poppelbaum, Principal Investigator.)

Summary

The remote APE power supply has been changed from RF to operate in the infra-red portion of the spectrum: Yiu Wo provides the details. Panigrahi's PENTECOST report deals mainly with final designs of the circuits to operate the special two-color tube. Jim Cutler describes the bundle arithmetic unit to be used in the Ergodic project. In the report on Telemaze, Ed Pott discusses some functions and designs associated with the "local" control, as well as the communication system. Finally, Mohammed El-Sonni, presents some preliminary findings in a new project, INCOM (INTERFACE COMputer) which investigates a particular aspect of the hardware - software interface, namely, the machine recognition of hand-drawn symbols.

M. Faiman - Ed.

1.1 APE (Project No. 25)

1.1.1 The Remote Power Supply

The APE machine is equipped with a remote power supply for the APEs. The power is sent to the APE remotely, so as to free it completely from any physical connection with other parts of the APE machine and therefore to further enhance the structural flexibility of the APE machine. The general requirements of the remote power source are: 1) The available power to each APE placed in the active region of the power source should be about 100 mW and 2) The operation of the remote power supply must not interfere with the operations of the APE.

This part of the APE machine was investigated originally by another graduate student in the Computer Hardware Group. The possibility of sending power to the APEs over a microwave frequency channel was first considered. An experimental set-up having a 60-watt transmitter at 1296 MHz with a helical transmitting antenna was built. It was found that almost sufficient power for operating an APE could be delivered to the APE if it was equipped with a quarter wave slotted line antenna for the reception of power and placed a few feet from the transmitting antenna in the direction of the main lobe of the radiation pattern. However, it was also found that the transmission of the microwave power interfered with the operations of the APEs. Each APE is equipped with three receivers operating at frequencies between 15 and 42.5 MHz. The transistors employed in those receivers have necessarily a high gain-bandwidth product. Unfortunately, these transistors also respond to RF signal at 1296 MHz. Without elaborate shielding and feed-through filtering, the leakage of the 1296 MHz RF signal into the box housing the APE circuits is

sufficient to upset the operations of the receivers. Some experiments have been conducted to house the APE circuitry inside an electromagnetically shielded box with 1296 MHz resonant traps guarding all feed-through terminals feeding data signals into the APE circuits. With careful tuning of the 1296 MHz resonant traps, the interference could be reduced to an acceptable level. However, such tuning is quite critical and could be detuned easily by coupling with nearby objects. Hence, this approach is not employed for remotely powering the APEs.

A more suitable solution to the remote powering of the APE is by means of solar cells. This approach eliminates the harmful RF interference. These solar cells are constructed with n-type silicon base material. A very thin layer of p-type material is diffused into the n-type base to produce a p-n junction with large area. When this p-n junction is short-circuited in darkness, no steady current will flow in the external circuit in spite of the existence of the contact potential of the p-n junction, as expected on thermodynamic grounds. However, if light in a suitable range of wavelength is allowed to fall on the p-n junction, a voltage is developed across the external circuit and current starts to flow with the terminal characteristic shown in Figure 1. Figure 2 shows the spectral response of the solar cells. For powering the APE, four solar modules are packed together on the top surface of the APE to absorb power from an array of incandescent lamps.

Yiu Wo

1.2 PENTECOST (Project No. 31)

1.2.1 Summary

Work on the modification of the receiver circuits for accommodating the Penetron cathode ray tube is underway. One of the main problems in

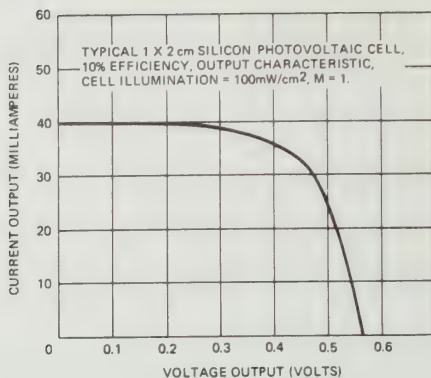


Figure 1. Typical Terminal Characteristics of a Photocell

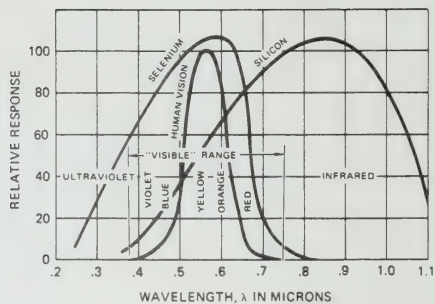


Figure 2. Spectral Response of Photocells

modifying the receiver circuits is to properly interface the solid-state circuits with the existing tube circuits. The vertical size control circuits and the video switches have been designed and tested in this quarter. They are described in detail below. Work on the horizontal control, brightness control and camera synchronization circuits will be completed in the coming quarter.

1.2.2 Vertical Size Control

Since the beam voltage is switched between 11 kV and 16 kV, the vertical and horizontal deflection signals must be changed every field so as to take care of the change in deflection sensitivity. The change in deflection sensitivity is given by the ratio $\sqrt{16/11}$. There are many ways of doing this. One method is to build a switched gain amplifier of good stability. This approach was discarded because of the difficulty in modifying the existing tube-type vertical amplifier. Instead, it was decided to switch the input signal to this amplifier by employing two analog gates as in an analog multiplexer.

The vertical size control circuit is shown in Figure 1. The direct coupled emitter followers isolate the previous vertical tube circuits and bring down the output impedance of the analog source. This enables the analog source to drive the high resistance load through the gate and minimizes the error introduced by the analog switch. Four matched silicon diodes (1N4148) form a diode bridge. Two transistor constant current sources T_1 , T_2 in their common base connection supply the current for the bridge. The resistor network biases the zener diode so that transistor T_3 can be conveniently controlled by the gate voltage. The 220 k resistors are used

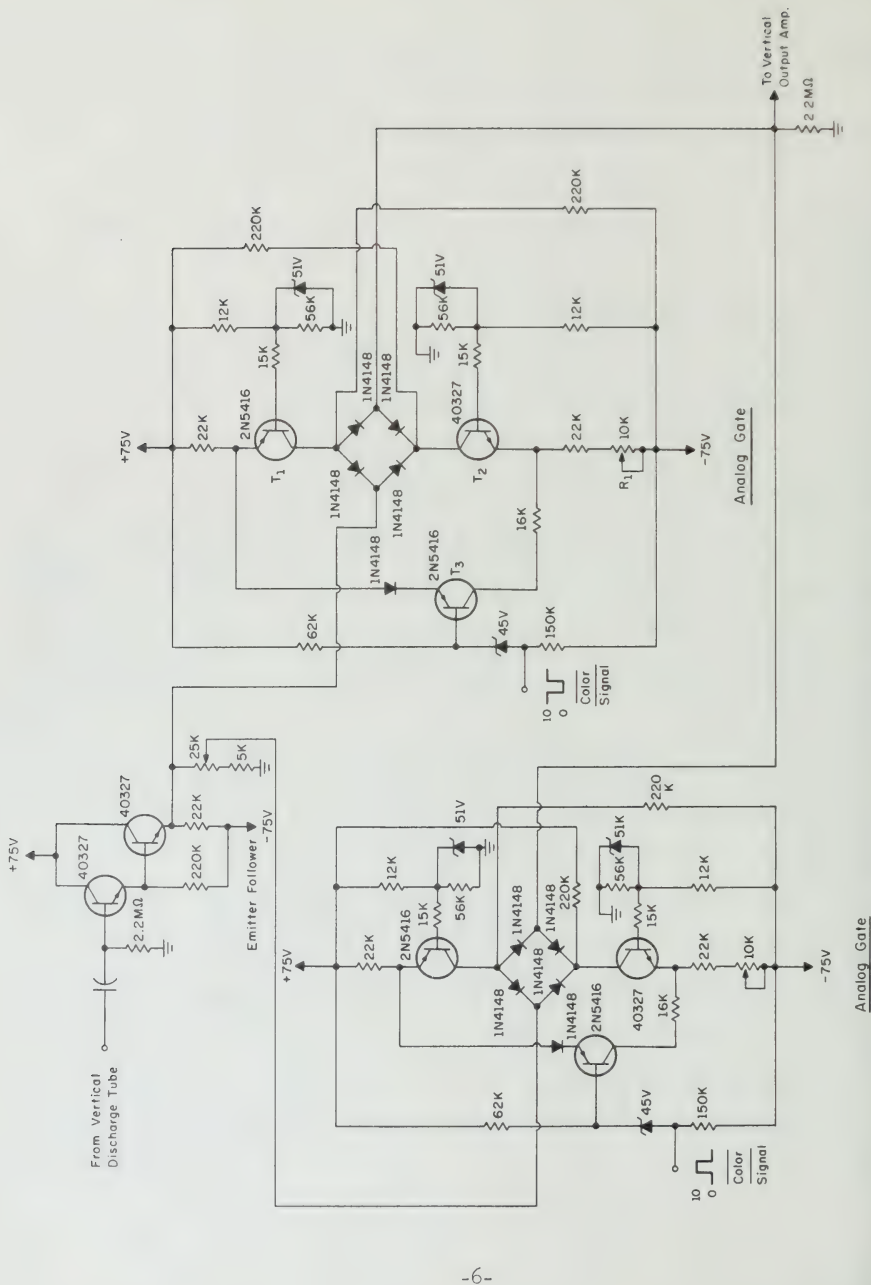


Figure 1. Vertical Size Control Circuit

to back-bias the diode bridge when T_1 and T_2 are off. This analog gate can handle analog signals of magnitude $\pm 50V$, is fast and has negligible offset voltage for our purpose.

The color signal is obtained from a multivibrator which is triggered by a negative going pulse derived from the sync separator and the integrator. This is shown in Figure 2. T_1 and T_2 amplify the composite video to drive the sync clipper T_3 . T_1 and T_2 are biased such that they almost eliminate the video. The clipped sync is applied to phase splitter T_4 . The emitter output is sent through a vertical integrator and then to an emitter follower that drives the multivibrator.

1.2.3 Video Analog Switch

To take care of the change in the beam voltage, the video signal amplitude has to be switched. This is shown in Figure 3. A voltage follower using a fast operational amplifier ($\mu A 715C$) isolates the previous video stage and has low output impedance to drive the FET analog gate. The latter has little offset voltage and the driving source is designed with a non-saturated transistor, so that this switch is capable of being operated at very high speed. It can handle input analog voltages of $\pm 15V$. This video multiplexer has been tested in the monitor and is able to switch the video signal for alternate fields.

1.2.4 Horizontal Deflection Control

Like the vertical and video signals, the horizontal deflection signal must also be switched. However, if the high-voltage output of the monitor is used as the constant high-voltage supply for the High Voltage Switch, we cannot switch the deflection signal since that would change the output

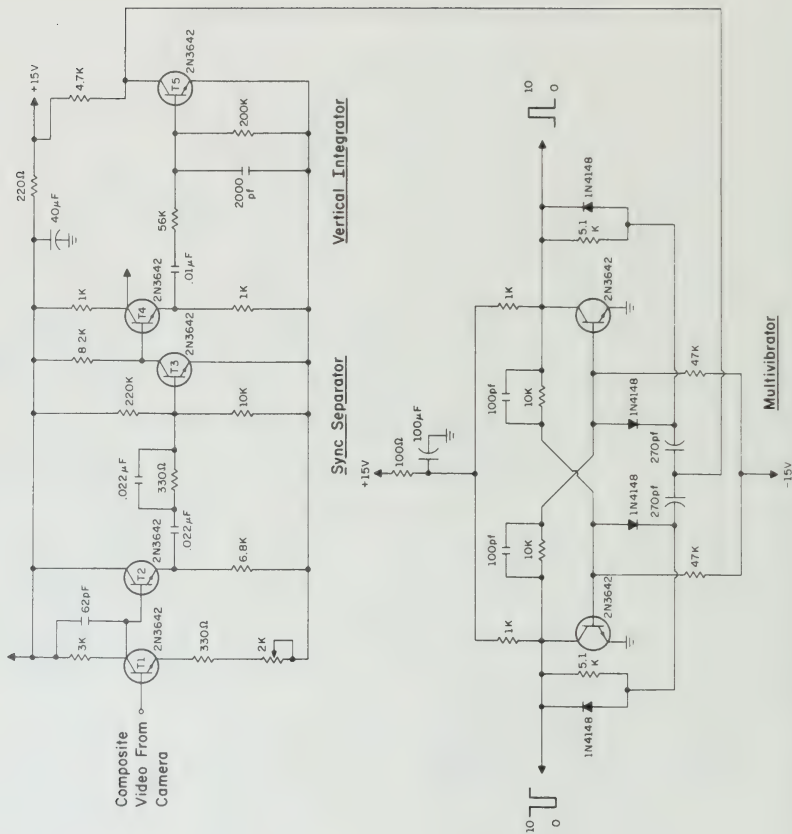


Figure 2. Vertical Blank Edge Separator and Multivibrator

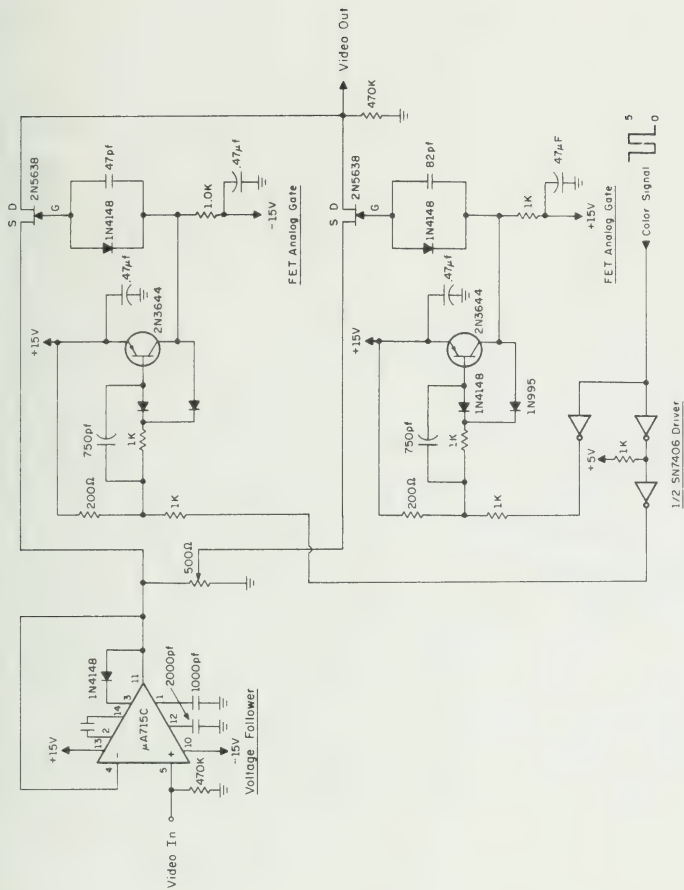


Figure 3. Video Analog Switch

voltage. Consideration was given to putting a gate-controlled resistance-pad across the deflection coil. But that would destroy the tuning of the horizontal circuit. Another possibility would be to employ a set of width control coils in parallel and series that could be switched in and out of the circuit such that the yoke current could be switched but the load to the flyback would remain constant. That would keep the high voltage constant. Subsequently, this approach was discarded for reasons of difficulty in getting proper coils. It was decided that a separate HV supply would be used. Now, the input to the horizontal amplifier will be controlled to change the horizontal size. A supply has been ordered from CPS Inc.

G. Panigrahi

1.3 Ergodic (Project No. 39)

1.3.1 Summary

Both of the generators and the processor of Ergodic have been completed this past quarter. The remaining part, the arithmetic unit, has been started, and it is expected to be completed by the next quarter.

1.3.2 Arithmetic Unit

The arithmetic unit does arithmetic operations on two ergodic bundles with the result also being an ergodic bundle. The operations are going to be addition, subtraction and multiplication.

Before going into the design of the arithmetic unit, a discussion of the number system is required. Numbers that are represented in an ergodic bundle are fractions from 0 to 1. In order to represent negative numbers a mapping that takes numbers from 0 to 1 into numbers from -1 to 1 is used.

That mapping is:

$$y = 2x - 1$$

where x = bundle representation ($0 \leq x \leq 1$),

y = actual number representation.

For example if 20 wires are energized in a 64-wire ergodic bundle at any one time, then the number represented is $-3/8$. If a second bundle has 40 wires energized, this represents the number $1/4$. Adding these two numbers together one obtains the fraction $-1/8$, or the resulting ergodic bundle should have 28 wires energized.

Figure 1 shows the block diagram of the arithmetic unit. The 64-bit shift registers store the two bundles. The sorters, which are also 64-bit shift registers, arrange the 1s and 0s so that 1) by looking at only one bit the signs of the numbers are known and 2) by doing circular shifting in the sorters (after the sort is completed and the signs are determined) the resultant bundle will then be ergodic.

Knowing the signs of the numbers and the operation to be accomplished, the control lines of the arithmetic sub-units can be determined. The circuit diagram of the arithmetic sub-unit is shown in Figure 2. Control line X determines whether A_i or \bar{A}_i are gated into the arithmetic gates and control line Y determines whether B_i or \bar{B}_i are gated into the arithmetic gates. Control lines K and L determine which output of the arithmetic gates is going to be connected to R_i . OV1 and OV2 enable or disable the overflow gates. Only three gates (per pair of input wires) are necessary to do all of the arithmetic operations: an OR gate, an exclusive OR gate and an AND gate. Table 1 shows the inputs and gates necessary to do the arithmetic operations.

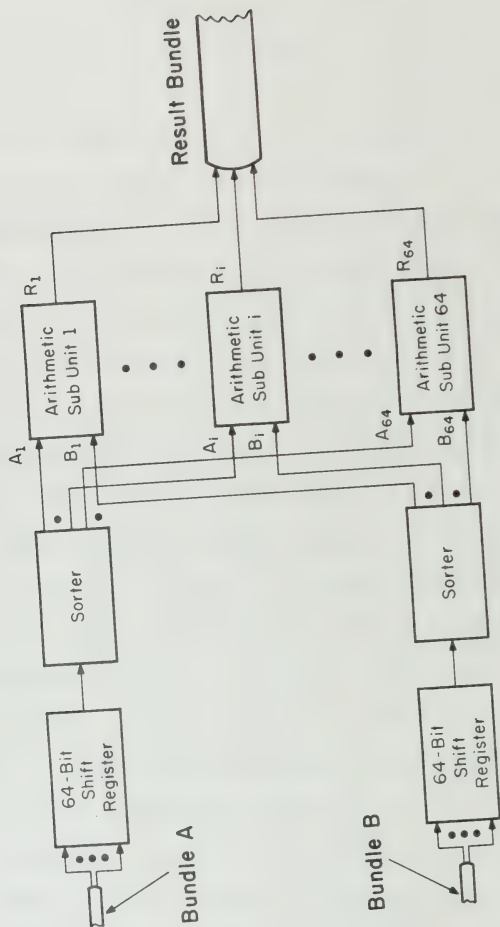


Figure 1. Block diagram of the Arithmetic Unit

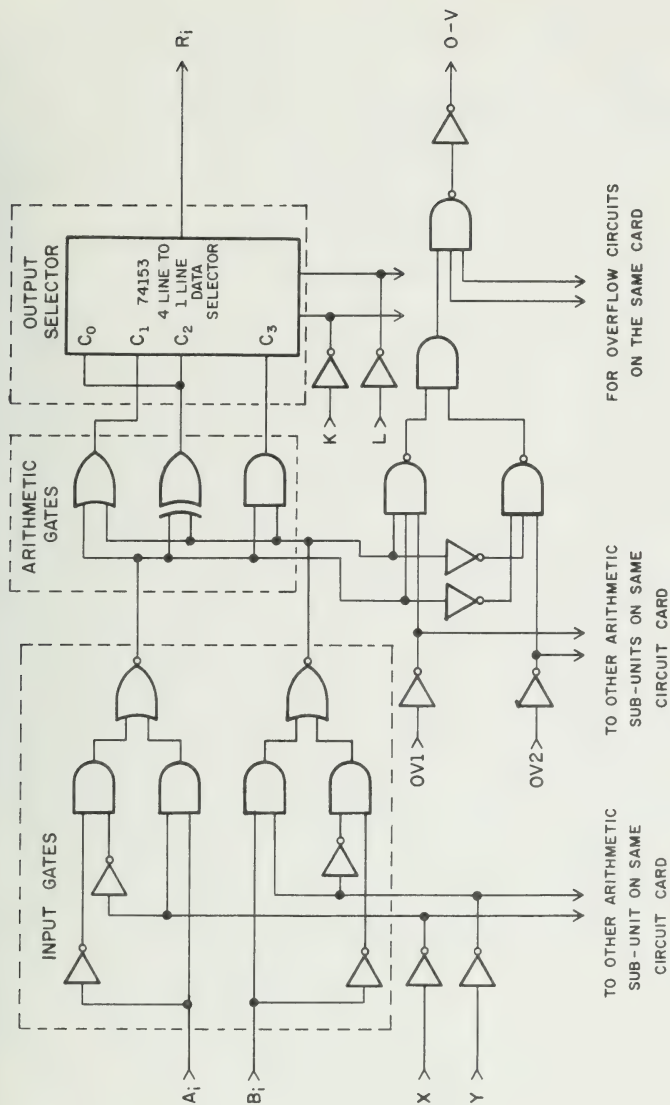


Figure 2. Circuit Diagram of an Arithmetic Sub-Unit

	Multiplication	Addition	Subtraction	
			A - B	E - A
A & B pos.	$R_i = \bar{A}_i \oplus B_i$ overflow disabled	$R_i = A_i \vee B_i$ overflow if $A_i \cdot B_i = 1$ for any $1 \leq i \leq 64$ (first 32 bits of A are first loaded with zero's)	Same as addition if A pos. & B neg.	Same as addition if A neg. & B pos.
A neg. & B pos.		$R_i = \bar{A}_i \oplus B_i$ overflow disabled (last 32 bits of A are first loaded with one's)	Same as addition if A & B are neg.	Same as addition if A & B are pos.
A pos. & B neg.		$R_i = A_i \oplus \bar{B}_i$ overflow disabled (last 32 bits of B are first loaded with one's)	Same as addition if A & B are pos.	Same as addition if A & B are neg.
A & B neg.		$C_i = A_i \cdot B_i$ overflow if $\bar{A}_i \cdot \bar{B}_i = 1$ for any $1 \leq i \leq 64$ (last 32 bits of A are first loaded with one's)	Same as addition if A neg. & B pos.	Same as addition if A pos & B neg.

Table 1. List of Arithmetic Operations

There are several details to be worked out that are concerned with the sort of the arithmetic unit. The question is how best to do the sort and arrange the 1s and 0s in the 64-bit shift register in the sorter. This, of course, depends upon the operation, and these details will be explained in the next report.

Jim Cutler

1.4 Telemaze (Project No. 41)

1.4.1 Project Summary

The object of TELEMAZE is to develop a simple system with an adjustable feedback delay, which could control the trajectory of a distant vehicle. The condition that is characteristic of this system is that the feedback delay time is much greater than times associated with the vehicle's movement. In this system control of the vehicle is obtained by steering an "image" of the vehicle through a "model maze" at the local station.

1.4.2 Major State Generator

The primary work carried out to this point has been to design the appropriate circuitry necessary to realize specific functions. To provide for proper sequencing of these parts, a "major state" generator is under construction. Changing from one state to another will be completed by pressing switches on the "mission control" panel. The logic circuits being constructed are based on a "real world" model as shown in the state diagram, Figure 1.

1.4.3 Video Gray Scale Generator

The function of this circuitry is to take two bits of information and convert this into four unique video levels for display on the local station television monitor. The two bits of information are from 1) the MAZE

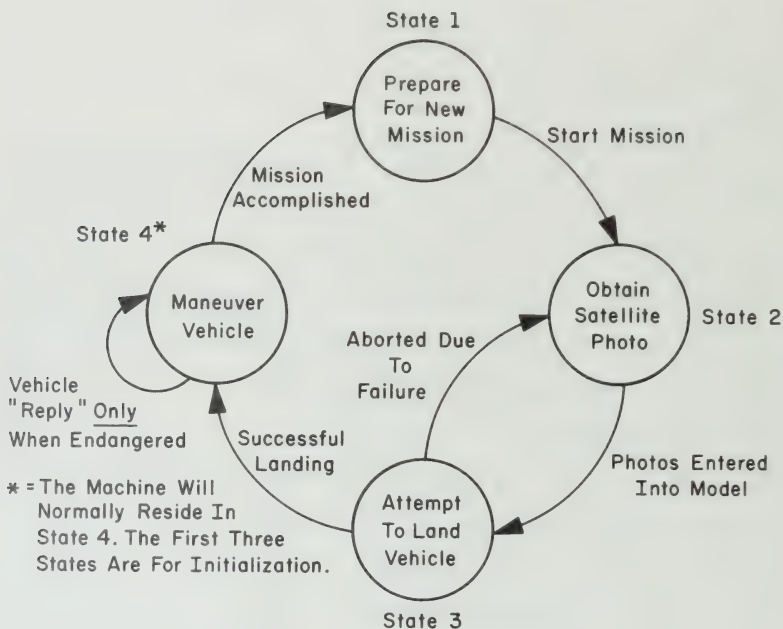


Figure 1. Telemaze "Real World" Model (Major State Generator)

memory and 2) the PATH memory. The MAZE information represents the "model image" in which our vehicle is steered. It is desired to have the obstacles denoted by the black level and unobstructed regions are white. PATH information represents the actual trajectory the vehicle has taken. The only remaining choice for display of the PATH is by using gray levels.

The actual circuit operation is described as follows (see Figure 2). MAZE and PATH information is decoded and sent to one of three driver circuits. Each driver circuit, when enabled to conduct, will cause the voltage at that point to approach zero.

When all drivers are off, the video output is about five diode voltage drops or about two volts. The other extreme is when the blanking driver is conducting and a voltage close to zero is obtained. The gray levels are then obtained by selecting the number of diode drops desired. This circuit provides a very fast response time which must be present in a video circuit.

1.4.4 Communication System

The basic function of this system is to transmit and receive information at both the local and remote stations (see Figure 3). The local station must send out trajectory information which the vehicle will attempt to follow. Only if it cannot complete this request will the remote vehicle generate a reply.

This information will take some delay time (T_D) for the transmitted radio signal to reach this distant vehicle. It will also take T_D seconds for the remote vehicle to make any necessary replies. Therefore it takes $2T_D$ seconds for a response to any request.

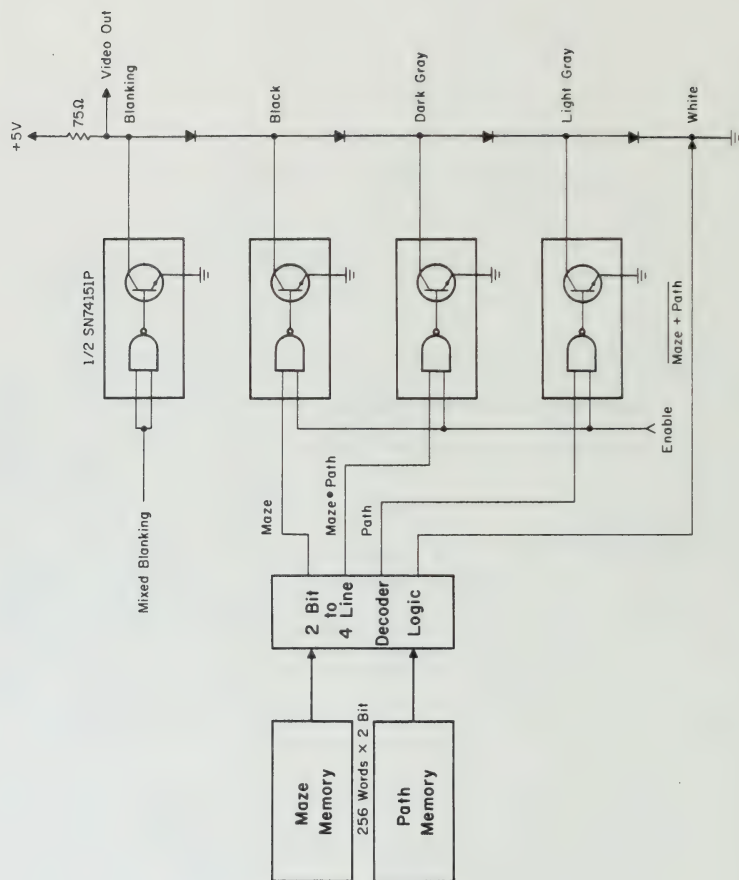


Figure 2. Video Gray Scale Generator

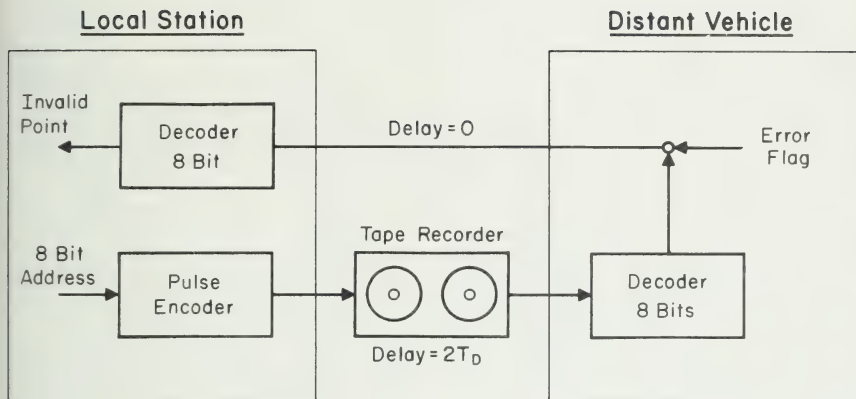


Figure 3. Communication System Block Diagram

To provide a realizable solution, this equivalent system with one tape delay unit will be used. The tape delay unit selected offers quantized delay times. The delay unit is equipped to use standard cartridges of the continuous loop variety. By selection of different tape lengths within the cartridge, tape delays of from 1 second to 30 minutes can be realized.

Edward J. Pott

1.5 INCOM (Project No. 46)

1.5.1 Introduction

This is a new project (INTERface COMputer) whose overall goal is to examine the hardware-software tradeoff at a computer interface where graphical information is to be input, e.g., from light pen and CRT or tablet. The major problem to be tackled is how best to "recognize" hand-drawn symbols.

One approach to this problem is indicated schematically in Figure 1. An input mechanism is connected to a computer via a piece of hardware, configured as a pluggable unit, that is designed to be capable of recognizing a specific class of hand-drawn symbols. Examples of such classes are 1) the set of alphanumeric and special characters commonly used in program writing or 2) a set of symbols and connectives typical of electronic circuits and so forth. Such recognition has frequently been performed in software, but the process is often consuming of both time and memory space. In INCOM this task is to be accomplished by hardware. A particular recognition processor would be plugged in, appropriate to the job in hand. Phase I of this project will design a small number of such pluggable units, with the aim of determining 1) the per-unit cost, and more important, 2) those aspects of the recognition process that are common to several units. The preliminary work on this project deals with the design of a recognizer for alphanumeric information.

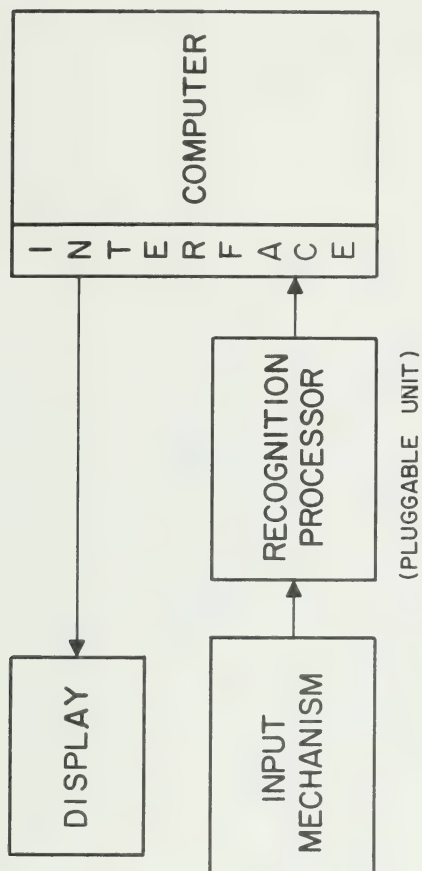


Figure 1. INCOM Phase I Block Diagram

1.5.2 A Nodal Analysis Technique

A handprinted character can be represented as a linear graph (nodes and edges). The nodes of this graph with their special characteristics and their spatial relationships must be taken into account in any nodal analysis. These nodes must be chosen such that maximum possible data compression can be achieved. The nodal analysis must also have a means of generating and choosing nodes which are sufficient to classify different characters in different classes.

We will apply this technique first to uniformly handprinted characters and then to characters not uniformly drawn.

Let us characterize the nodes as follows:

T node = A terminal node connected to one edge only.

L node = A node connected to two edges pointing to the left.

R node = A node connected to two edges pointing to the right.

N node = A node connected to more than two edges.

U node = A node connected to two edges pointing upward (which would be the first L, R or N node by scanning vertically).

D node = A node connected to two edges pointing downward (which would be the last L, R or N node by scanning vertically).

Figure 2 shows examples of the various node types.

Now each character is actually a collection of nodes (T, L, R, N, U, D) with some spatial relationship which distinguishes this character from the others. By scanning these nodes vertically and/or horizontally (from left to right) we can get a representation of the character by a sequence of nodes.

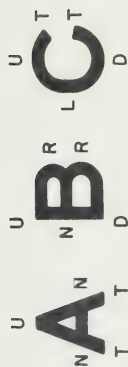


Figure 2. Examples of T, L, R, N, U and D Node Types

Some characters can be uniquely represented by vertical scanning of the nodes by both vertical and horizontal scanning. Thus, using the example of Figure 2, and denoting vertical and horizontal scanning by subscripts v and h, respectively:

$$A_v = U, N, N, T, T$$

$$A_h = T, N, U, N, T$$

$$B_v = U, R, N, R, D$$

$$B_{nh} = U, N, D, R, R$$

$$C_v = V, T, L, T, D$$


$$C_{nh} = L, V, D, T, T$$

When uniformly handprinted characters are classified in this way, it is found that only two characters (K and X) need horizontal scanning to distinguish between them while vertical scanning is sufficient for the rest of the characters.

1.5.3 Dealing with Nonuniformity

Any nodal analysis technique cannot be considered complete unless it is able to deal with most of the nonuniformity of writing. Some of these nonuniformities come from the style of writing; others are due to input discontinuity and noise. In the proposed technique some of the nonuniformities are taken care of inherently. Let us look at the character A as an example. We have seen that $A_v = U, N, N, T, T$. This description is the same for the following nonuniformly drawn characters representing 'A'.



For the problem of nonuniformity which results from discontinuity we suggest the merging of superfluous nodes.  is considered an 'A', and we apply

vertical scanning of the nodes, we find T, T, N, N, T, T which is the same as $H_v = T, T, N, N, T, T$. But the top nodes T and T are "near enough" for an 'A', (in relation to the overall linear dimensions of the character); so we can merge them together to obtain a 'U node'.

An extension of the merging technique can be applied to "noise" breaks and to cutting down "overdrawn" strokes. Further details will appear in subsequent reports.

M. El-Sonni

2. HARDWARE SYSTEMS RESEARCH

(Supported in part by the Atomic Energy Commission under Contract US AEC AT(11-1) 1469, W. J. Poppelbaum, Principal Investigator.)

Summary

The LASCOT deflection system is now operating satisfactorily in the green (Cooper). Doug Sand describes new components that have been incorporated into OLFT. Sematrix is complete, awaiting only a computer to exercise it. Dick Blandford reports on LINDA's new method of scanning picture segments and changes in the sweep mechanism. Stereomatrix has had the precision of its coefficient generator extended by three bits, courtesy of Steve Whiteside, who also mentions problems with the laser display. Sik Yuen describes how the Scantrix display elements are driven. An update of the FROG simulator and the design of the T* element are discussed by Debasish Bose. Martin Jer has a redesign for parts of BEACON. Bob Budzinski presents some new circuits for the OCOMO project. Finally, Martin Jer reports some preliminary work on a new project, Caecotron (a tube for the blind), which is based on the technique of converting a visual, TV-like scan of a scene into a varying audible tone.

M. Faiman - Ed.

2.1 LASCOT (Project No. 09)

Horizontal and vertical oscillator outputs were taken from a Zenith Model C3520 television set and were used to drive horizontal and vertical mirrors. The relative phase shift between the video signal and the mirror deflection was nulled by introducing delays (monostable multivibrators) in the path of the deflection waveform. A raster was obtained.

The video and chroma signals were also tapped from the television set. A video signal was added to the chroma signal and was used as an input to the EOLM driver. A picture of reasonable contrast and about 150 spots reduction was obtained for the green color.

Work is under way to extend the same to the remaining two colors.

Mehernosh Cooper

2.2 OLFT (Project No. 12)

The major project for this quarter has been the Ph.D. thesis for the COLFTAR system. This thesis will be released during the next quarter. In addition, several system modifications and repairs were initiated. First, two new 5 mil KD*P crystal assemblies were obtained from Isomet, and one was installed in the light valve to replace the 10 mil assembly, which was severely cracked by transient "overwrites". Second, new write and erase electron guns were obtained from Westinghouse and installed. The write gun has a new "coated-powder" cathode which is expected to be a substantial improvement over the old tri-carbonate cathode, providing higher beam currents, longer life, and less deterioration due to up-to-air cycles. Operation of the light valve can resume after the vacuum pumps have been thoroughly cleaned, which has become necessary because of surface saturation of the ion-pump electrodes, which typically occurs after many months of operation.

Douglas Sand

2.3 Semantrix (Project No. 24)

During this quarter the hardware was completed, and a departmental report is being issued giving the system description.

Trevor Mudge

2.4 LINDA (Project No. 28)

2.4.1 Summary

Project LINDA (LINE Drawing Analyzer) is an attempt to produce a machine capable of recognizing a small vocabulary of simple line drawings made up for the most part of circles and polygons. Recognition is to be independent of orientation or size within reasonable limits. Recognition is based on the idea that if one is able to determine the simple parts of more complex figures and their general place relationship, one can make an accurate guess as to what the figure is. A more detailed description of LINDA may be found in the quarterly report for Jan., Feb., and March, 1972 including a system block diagram.

2.4.2 Project Status

As the project now stands, LINDA is being constructed on a simple scale and more complex features are added as a more thorough understanding of the problems of the simpler scale is reached. To date, LINDA is able to work with line drawings having up to three parts in a vertical line. The most pressing problem that this simple model has made known is that of noise in the analysis of the video signal. During the past quarter several things have been changed to contend with this problem. The main change was to eliminate the photocell line in favor of a faster method.

Although the photocell line was simple and graphic and even though the resolution problems it created were largely solved through the

use of fiber optics (see the report for July, Aug., and Sept., 1971) , it was very slow and time integration over several photocell line sweeps was not practical. It will be recalled that the purpose of the photocell line was to sample the shaded figure continuously as it passed under the photocells. The sum of all the photocell signals was then found. This sum signal contained discontinuities corresponding to discontinuities in the figure, making polygon identification a simple matter. In place of the photocell line the sum signal is now created by chopping the video on each line and doing a D/A conversion. It can be seen that with this method several sum signals can be examined and compared in the time it previously took to generate one sum signal (about 0.2 second). The effect of comparing several sum signals is to eliminate random noise in the system. This method also has the welcome fringe benefit of allowing the system to use the full resolution that a TV frame has to offer.

One other change of importance made during the last quarter was to alter the vertical sweep rate of the flying spot scanner so that it now takes twice as long to make one vertical sweep. Each frame now contains only one field of 500 lines instead of the conventional even and odd interlaced 250 line fields. The bad effect of this change was to produce a slight flicker in the display. This is not as bad as one might at first think since no moving pictures are being displayed. The good effect of this change was to double the vertical resolution of the summing signal. This is due to the fact that the sum signal is taken over one field only which now has 500 lines instead of 250.

Other less important changes made during the last quarter include building new sweep circuits and improving the video pulse amplifier. These circuits will be discussed in subsequent reports when cards and testing are complete.

2.4.3 Future Work

It is hoped that noise problems are largely taken care of and the next quarter can be devoted to developing a scheme to separate more complex figures into simple parts.

Dick Blandford

2.5 Stereomatrix (Project No. 30)

2.5.1 Changes to Coefficient Generator

The coefficient generator was extended from nine bits to twelve bits plus sign. This involved adding packages to the twelve coefficient storage registers, adding a new four-bit gating card, and modifying the multiplier control to do twelve steps rather than nine. Fortunately the existing adder was designed for twelve bits plus sign, so modification has simply meant bringing out the three low order bits.

The four-bit gate design is given in Figure 1. Control of this gate was simplified by using the ABCD sequence commands directly from the control card. These commands generate the twelve coefficient selection pulses on the control card. The previous design used all twelve pulses on twelve wires. It was also observed that the coefficients are gated always in the same order, but the data to the sine multiplier is shifted by one for y-rotation and by two for z-rotation. This is accomplished with the 7483 adder package. The strobe inputs of the 74150's are used to stop data output when no computations

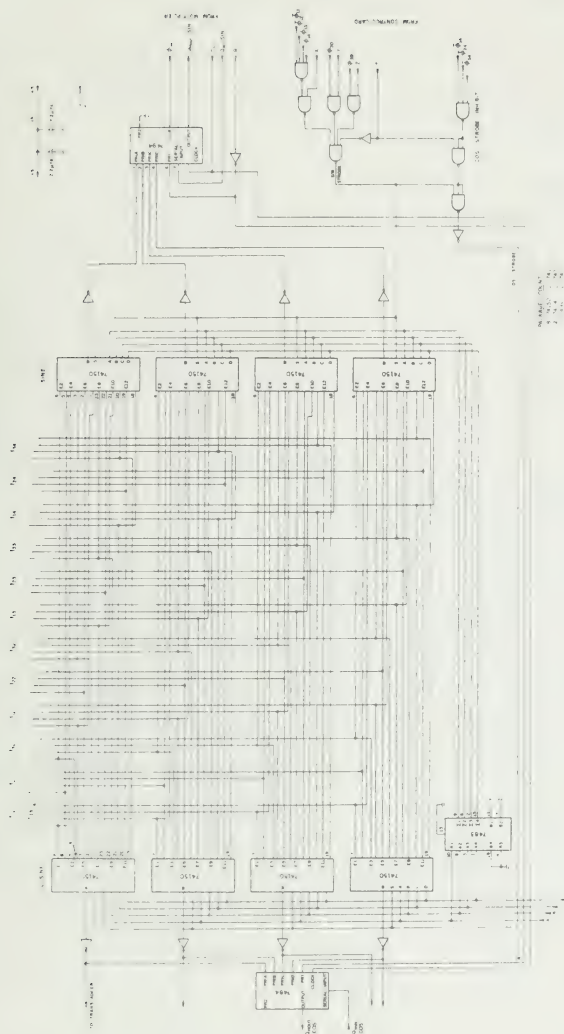


Figure 1. Four-bit Gate Design

are needed. A step-by-step check of the computations has yet to be accomplished, although the circuit seems to be working.

With the increased accuracy (a factor of 8), the angular increment was decreased from 10° to 1° . This causes some distortion, and a 2° increment may be more acceptable.

2.5.2 Display

Parts for the laser power supply were finally received in late August. The supply is working with the new start circuit and added power transistors. However, some of the transistors seem to be heating more than others, and there are some undesirable audible oscillations from the supply. It may actually be operating in a switching mode. This is to be corrected.

Steve Whiteside

2.6 Scantrix (Project No. 35)

2.6.1 Enabling the Display

During the past quarter, work was emphasized on the display unit. A modified version of the display element, which comprises essentially the LED and its driving circuit, is shown in Figure 1. Normally, the voltage at point A of the circuit is at 10 volts. When sampling, the voltage at A is lowered 5 volts through a switching network as shown in Figure 2. The voltage at B subsequently follows, dropping to zero and allowing the diode D to be turned on. An analog input signal of zero to five volts is then held by capacitor C, which turns on the LED through the FET. The RC constant is made long enough for the stored voltage to decay in approximately 16 msec, after which the cycle repeats.

In the system, there are 4096 such units arranged in 64 rows of 64 elements each. Figure 3 shows the block diagram of the whole display

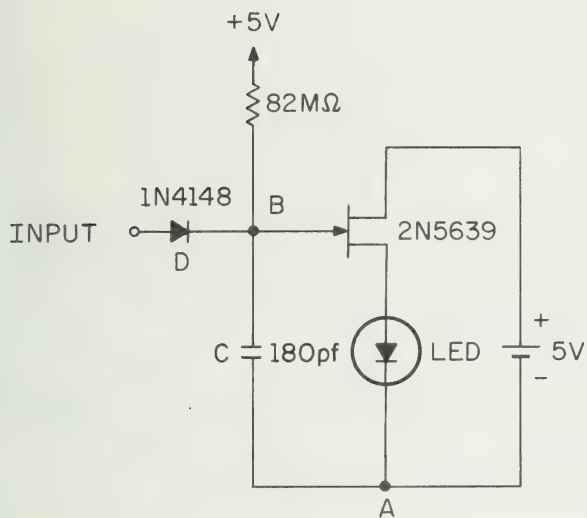


Figure 1. Display Element

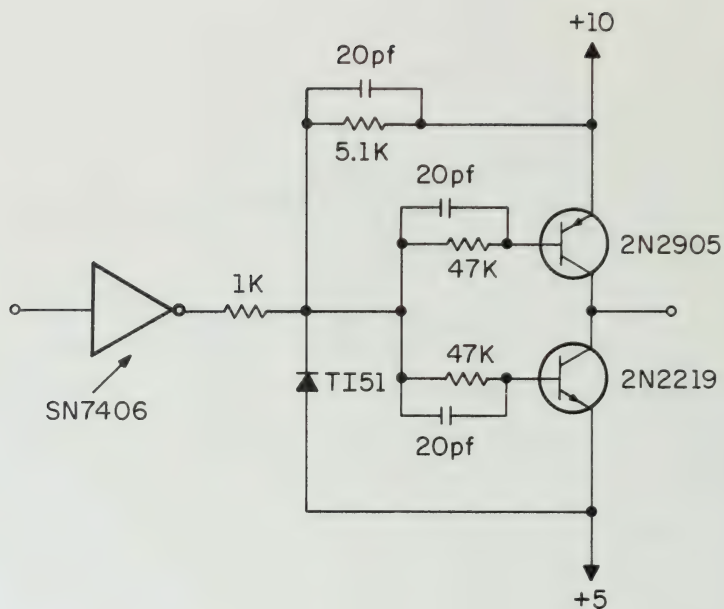


Figure 2. Inverting Circuit with Level Shifting

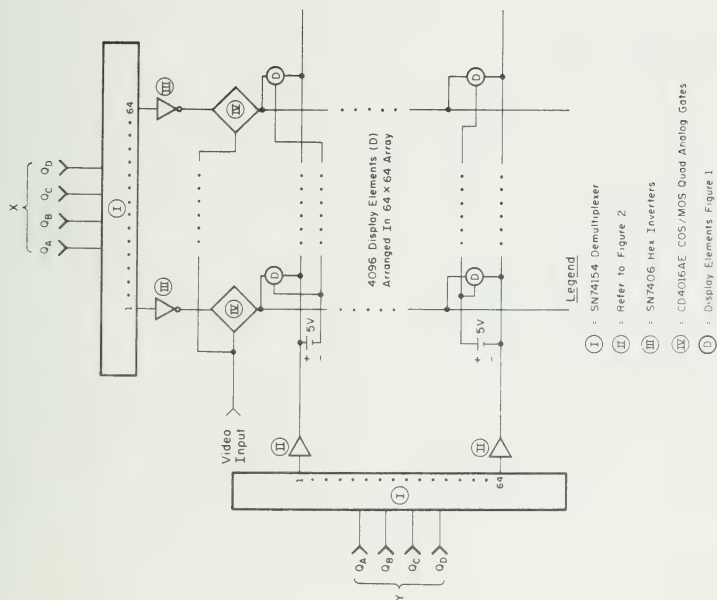


Figure 5. Scautrix display unit with X-Y Decoding

unit. The two units of four demultiplexers each form the horizontal (X) and vertical (Y) decoding circuits. The horizontal unit is responsible for switching the COS/MOS analog gates on and off, one at a time, along each row. With a certain Y input, the corresponding demultiplexer output goes to zero. This triggers the circuit in Figure 2 to switch the whole row of 64 display elements up to five volts, thus allowing inputs signals to appear on the display elements via the analog gates.

Sik Transit Yuen

2.7 FROG (Project No. 36)

2.7.1 Simulation Study

The first simulation of FROG (reported in the last QTPR) showed some defects in the logic design of the basic building block, the T* element, of the World Model. These have now been rectified and the simulator will now be used for the final simulation. It is hoped that the results will be processed by the end of the next quarter.

2.7.2 Hardware Implementation

A preliminary design of the T* element using strictly digital processing is complete. But, at this point, it appears that stochastic processing would be better suited for implementing some of the logic of FROG. Presently a design of the T* element incorporating stochastic processing is in progress.

Debasish Bose

2.8 BEACON (Project No. 38)

2.8.1 Amplifying Cell

A different light amplifying cell is now being designed to replace the one described in the last quarterly report. This cell is capable of doing

two things: in the normal operating mode, the cell will amplify the signal coming in from the photo transistor linearly to drive a LED; while in the "freeze" mode, it is capable of converting the incoming signal to a two-level (on-off) display. There is also a threshold control so that the two-level decision can be changed by the different voltage levels applied to the reference input (Figure 1).

2.8.2 Sequential Freeze Circuit

It takes about 30 msec to display one frame of a regular TV picture. But due to the odd and even horizontal raster lines, the phosphor on the screen is "refreshed" every 15 msec. The photo transistor responds to this rapidly. The object here is to freeze a row of 32 cells after the horizontal raster line has scanned through the area that the phototransistor "sees". Since in 15 msec there are 262.5 raster lines, and each row of phototransistors sees 16 raster lines. The last 13.5 lines are ignored, i.e. not seen by any phototransistor.

The horizontal sync signal is taken from the TV monitor, through a one shot and a binary counter to divide it by 16 and is then used as a clock for the 74164. The input from a manual switch controls the display and freeze mode (Figure 2).

Martin Jer

2.9 OCOMO (Project No. 42)

2.9.1 Summary

OCOMO is a voice communication system using optical encoding and decoding. A voice signal is sampled, quantized and stored. This voltage is applied to the vertical deflection plates of a CRT. On the face of the

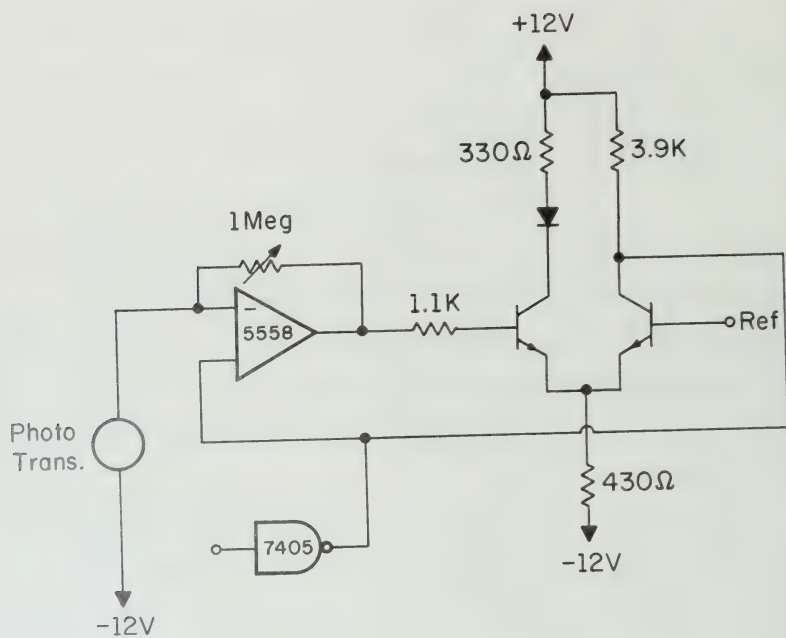


Figure 1. Amplifying Cell

CRT are film strips with opaque or transparent rectangles. The quantized voice voltage deflects the CRT beam to one of the strips. A horizontal sweep of the constant intensity CRT beam produces a sequence of light pulses which are detected by a photomultiplier tube. The output of this tube is then transmitted. The transmitted pulses modulate the beam of the CRT at the receiver. The horizontal sweep is synchronized to the one at the transmitter. The vertical sweep covers the entire vertical dimension during one bit of the transmitted pulse train. On the face of the receiver CRT are the complements of the strip at the transmitter. Thus for each code word sent one and only one film strip at the receiver will be the opposite of the one used to generate the transmitted pulse train. This opposite strip will pass no light during a horizontal sweep. This strip will be detected by a photocell array. Corresponding to the strip will be a voltage which will be applied to a loudspeaker.

2.9.2 Present Work

An audio amplifier with AGC has been designed and tested and is shown in Figure 1. An AGC control is used because the audio signal is quantized into one of 64 voltage levels. If the audio signal exceeds the largest voltage level, the quantized signal will be distorted.

The audio amplifier consists of a buffer stage and a stage with a voltage gain of 210. The AGC works by changing the dynamic impedance of the four diodes at the AGC input. The AGC output signal is rectified, filtered and is then used to set the bias of the diodes through an emitter-follower configuration. The output of the AGC is also amplified by the final stage so that the output never exceeds 10 volts peak to peak. The bandwidth of

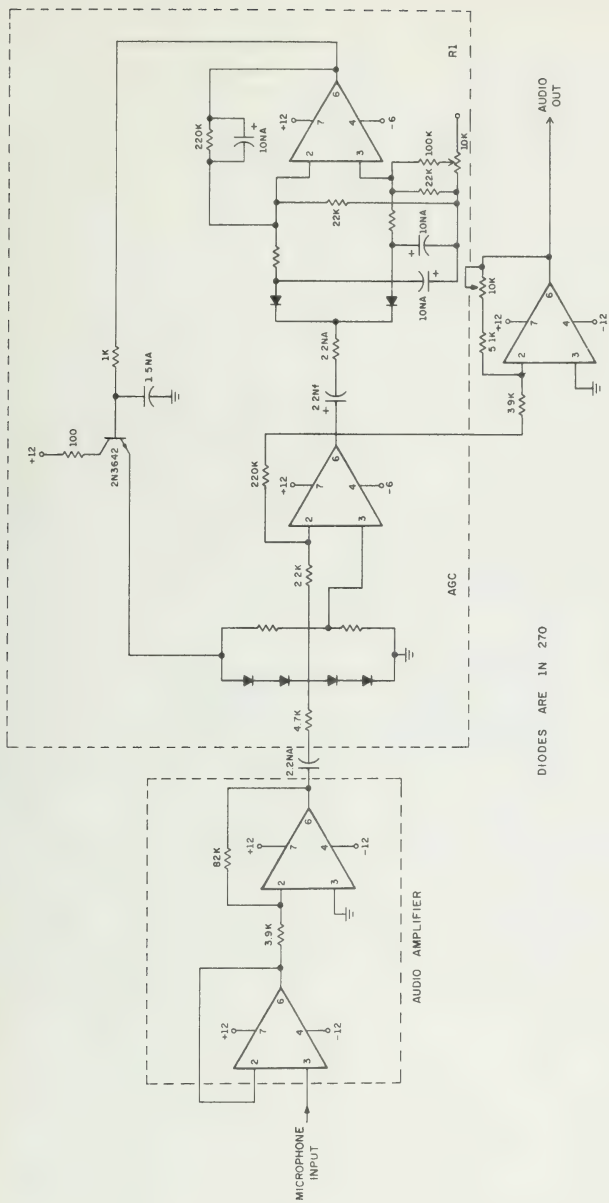


Figure 1. OCOMO Audio Amplifier



Figure 2. Sample, Quantize and Store Circuit

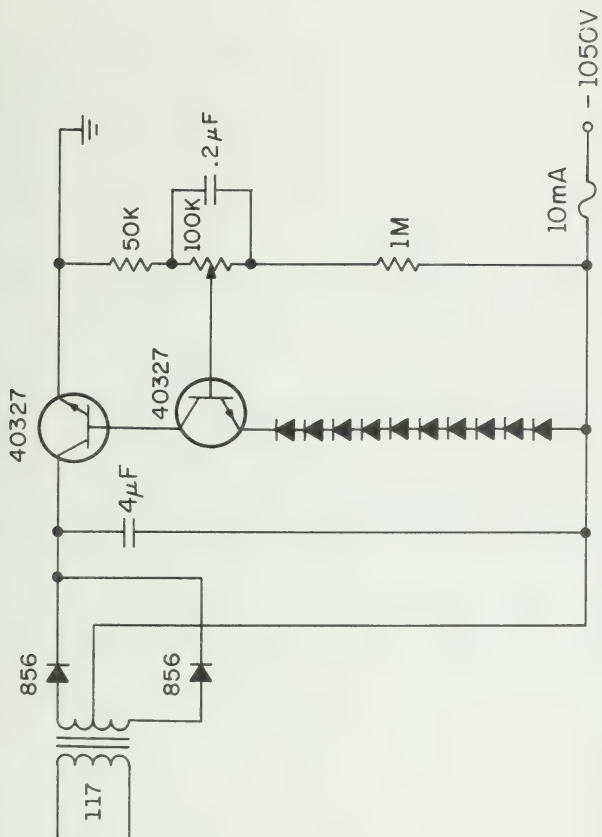


Figure 3. Photomultiplier Power Supply

this amplifier is 9 kHz. The output of the AGC exhibits a change of 6 db (2-fold increase) for 60 db (1000-fold increase) change at the output.

The sample, quantize and store circuit, shown in Figure 2, has also been designed and tested. The circuit has a tracking A/D converter which continuously digitally quantizes the analog signal. This digital representation is sampled and stored in the register. The output of the register is reconverted to an analog signal.

The input stage of the A/D converter is a level shifter and a buffer. The MC 1406 is a six-bit D/A converter and is used as a current sink. If the voltage at the current sink is $\pm \frac{1}{2}$ of a voltage step ($\frac{10}{64}$), the counter is inhibited. If the voltage is not within this range, the counter is either counting up or down at a 3 MHz rate.

A higher voltage power supply has been designed and built to operate the photomultiplier tube. Figure 3 shows the circuit diagram. The output voltage is adjustable between -950 and -1050 volts. The output has .15 volt peak to peak ripple at the required output current (5mA).

Robert Budzinski

2.10 Caecotron: A Tube for the Blind (Project No. 43)

During the last quarter, preliminary investigation into a new project called Caecotron was started. Caecotron, a tube for the blind, is a system that transforms and displays a television picture into aural form. Since the response of the human ear is considerably lower than that of the eye, the relative high resolution of a TV picture has to be drastically reduced to fit the bandwidth of the ear.

It was proposed that a 16-step tone sequence should be used to simulate the video picture, with each tone in the sequence lower in pitch

than the previous one. Each note, which is analogous to a TV line, is subdivided into 16 equal subdivisions, so that a slight decrease or increase in frequency at a particular spot would indicate the black or the white level. Such a scanning scheme is similar to that of commercial television except that there is no interlace.

In order to indicate depth, an electronic telemetry system using two cameras is used to control the intensity of the aural signal. An unusually loud note would, for example, indicate a nearby object. This scanning scheme is summarized in Figure 1.

Owing to the fact that not all the information available from a TV picture is utilized in the aural picture, a few interesting options are open. Instead of using every 32nd line from the TV picture, the operator has the capability of closing in on an object by selecting lines that are closer together and by scanning a smaller section of the picture. All this can be accomplished without zoom lenses.

Some experimentation in this direction had been performed during the last quarter. Telephone poles, tall and short fences have been simulated using a 16-tone generator and appropriate control logic, and results indicate that the above is a very possible approach to a tube for the blind.

Bernard Tse

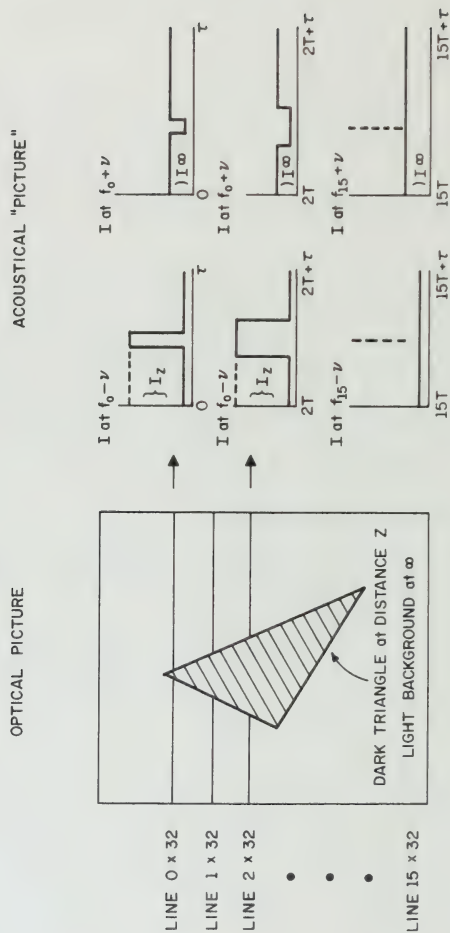


Figure 1. Example of a CAECOTRON Acoustical "Picture"

PUBLICATIONS

Shiv Prakash Verma, "Perspective Transformer for the Stereomatrix 3-D Display System", Department of Computer Science Report No. 532, University of Illinois, Urbana-Champaign, October 1972.

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. EC REPORT NO. C00-1469-0213	2. TITLE 3rd Quarterly Progress Report 1972
-----------------------------------	--

TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:
Title of conference _____
Date of conference _____
Exact location of conference _____
Sponsoring organization _____
☐ c. Other (Specify) _____

RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

REASON FOR RECOMMENDED RESTRICTIONS:

SUBMITTED BY: NAME AND POSITION (Please print or type)

W. J. Poppelbaum
Professor and Principal Investigator

M. Faiman
Assoc. Prof. of Comp. Sci.

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature

W. J. Poppelbaum

M. Faiman

Date

September 30, 1972

FOR AEC USE ONLY

AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION
RECOMMENDATION:

PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

3. SOFTWARE SYSTEMS RESEARCH

3.1 Numerical Processes

3.1.1 DIFSUB (R. L. Brown)

Two proposed changes to DIFSUB were investigated this quarter. The first involved the ability to use DIFSUB for both forward and backward integration (positive and negative time increments). This is accomplished by comparing DABS(H) instead of H with HMIN in three places. Several users requested this facility and have subsequently tested the resulting program finding that it works properly.

A more substantial change is a self-starting procedure using a fourth order Runge-Kutta (R-K) method to compute the value of all dependent variables at an initial step H beyond T_0 (initial value of independent variable T), subject to error control, and then continuing with either Adam's method or stiff method at third order.

Using the initial step size H given by the user-provided calling program, DIFSUB takes one R-K step. Restoring the initial values of all variables of integration, two steps of length $\hat{H} = H/2$ are taken. The error in the given method is

$$2*\hat{H}^5*\epsilon$$

for two steps of length \hat{H} , and

$$32\hat{H}^5*\epsilon$$

for one step of length $H = 2*\hat{H}$. By taking an appropriate norm of

$$|Y_{\hat{H}} - Y_{2\hat{H}}|$$

we have an estimate of $30\hat{H}^5*\epsilon = 30*ERR$ where ERR is the actual error. Using a "heuristic factor" to restrain the possible increase in step size due to ignoring round-off error, one computes

$$ERR = \frac{\|Y_{\hat{H}} - Y_{2\hat{H}}\|_2}{15}$$

and compares this with EPS, the user given error criterion. Clearly,

$$\frac{EPS}{ERR} = \left(\frac{H_N}{H}\right)^5$$

where H_N is the step size that would have given $ERR = EPS$ and is the optimum step size for the given error criterion. By computing

$$R = (EPS/ERR)**.2$$

we can tell whether or not the step should be repeated with a step size of $H_N = R*\hat{H}_0$. If $R > .9$, we know that our step is successful and no smaller step size is needed. However, if $R > 100$ the problem of round-off error due to too small a step size enters, so the step size is accepted if $.9 \leq R \leq 100$. Otherwise, a new step size $H = R*\hat{H}$ is chosen and the above is repeated. If three attempts fail to compute an acceptable set of values, further attempts are abandoned and the program continues using a first order multivalue method.

After completing the R-K step, six elements of data are available; the initial values Y_0, Y'_0 ; the intermediate values at $t = t_0 + \hat{H}$, $Y_{1/2}$ and $Y'_{1/2}$ and the final values Y_1 and Y'_1 at $t = t_0 + H$. From these one needs to obtain $Y(J,I)$ in DIFSUB, where

$$Y(J,I) = \frac{Y^{(J-1)} * \hat{H}^{(J-1)}}{(J-1)!} \text{ for the } I\text{-th}$$

dependent variable at $T = T_0 + H$ for J as large as possible with acceptable accuracy. Two methods were investigated for this procedure.

The first used the Taylor expansion represented in matrix form by

$$TY = Y_{RK}$$

where $Y = (Y(1,I), Y(2,I), \dots, Y(6,I))^t$ as above, and $Y_{RK} = (Y_0, Y'_0 * \hat{H}, Y'_{1/2}, Y'_{1/2} * \hat{H}, Y_1, Y'_1 * \hat{H})^t$, and T is the matrix, t is the transpose operator. T is then inverted and we compute

$$T^{-1} Y_{RK} = Y$$

However, since all but the first two elements of Y_{RK} may be in error by as much as $\pm EPS$ in the case of $Y_{1/2}$ and Y_1 and of $\pm EPS * H$ for $Y'_{1/2} * H$ and $Y'_1 * H$, the actual calculation (using worst case error analysis and ignoring round-off error) is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 7 & 1 & 16 & 8 & -23 & 6 \\ 34 & 5 & 32 & 32 & -66 & 13 \\ 52 & 8 & 16 & 40 & -68 & 12 \\ 24 & 4 & 0 & 16 & -41 & 4 \end{bmatrix} \begin{bmatrix} Y_0 \\ Y'_0 * H \\ Y_{1/2} + \epsilon \\ (Y'_{1/2} + \epsilon) * H \\ Y_1 + \epsilon \\ (Y'_1 + \epsilon) * H \end{bmatrix} = \begin{bmatrix} Y_1 + \epsilon \\ (Y_1^i + \epsilon) * H \\ (Y_1^{ii} * H^2 / 2) + \epsilon(39 + 14 * H) \\ (Y_1^{iii} * H^3 / 8) + \epsilon(98 + 45H) \\ (Y_1^{iv} * H^4 / 24) + \epsilon(84 + 52H) \\ (Y_1^v * H^5 / 120) + \epsilon(41 + 20H) \end{bmatrix}$$

Since $Y(J, I)$ becomes smaller as J increases because $Y(J, I)$ is proportional to $\frac{1}{(J-1)!}$ and to $H^{(J-1)}$ ($H < 1$) and $Y^{(J-1)} < Y^{(J)}$ (in general) for $J > 3$ the error terms due to the matrix multiplication can be larger than the true values and the calculation is useless.

An alternative method computes

$$Y'(t + H/4) = Y'_{1/4} = \frac{Y_{1/2} - Y_0}{H},$$

so

$$y'_{1/4} * H = y_{1/2} - y_0, y'_{3/4} * H = y_1 - y_{1/2}.$$

Letting $G_i = Y'_i * H$, $i = 0, 1/4, 1/2, 3/4, 1$, and applying Newton's divided difference method to G_i , realizing that the K -th divided difference of f ,

$$f[x_0, \dots, x_k] = \frac{f^{(K)}(\xi)}{K!},$$

for ξ in $[x_0, \dots, x_k]$ for ordered points x_i , and if the K -th derivative of f exists. In the case of multivalued methods, by replacing f by \hat{f} where \hat{f} is the polynomial that solves the equivalent difference equation used by the multivalued method to approximate the solution to the differential equation, the existence of $\hat{f}^{(K)}$ is assured. The following formulae for the $Y(J, I)$ and their error bounds result:

$$Y(3,I) = Y_1' * H + \epsilon * H - (Y_1 + \epsilon) + (Y_{1/2} + \epsilon)$$

$$Y(4,I) = 2/3 * (Y(3,I) + \epsilon(2 + H) + Y_{1/2} + \epsilon + Y_{1/2}' * H + \epsilon * H)$$

$$Y(5,I) = 2/3 * (2 * (Y_{1/2}' + \epsilon * H) + Y_0 - Y_1 + \epsilon + 1.5 * Y(4,I) + 2/3 \epsilon * (4 + 2H)).$$

The worst case truncation errors are therefore much smaller than in the Taylor expansion matrix case. Even though the higher order derivatives are evaluated on open intervals rather than at definite points, acceptable accuracy should be achieved.

Differential equations whose closed solutions are known and which have easily calculated n-th order derivatives (e^t , e^{-t} , $\sin t$) were used to test both methods and confirmed the predictions that the Taylor expansion method produced incorrect data. Fairly accurate answers were calculated using the Newton's divided difference method. Values for $Y(6,I)$ were often inaccurate, and for $Y(5,I)$ were off by a factor of less than 10. Because $Y(5,I)$ is used to calculate the best multivalued method step size, this is acceptable and $Y(1,I)$ through $Y(4,I)$ are used to start up the multivalued method at third order.

Since all that the program needs for this start up procedure is a set of initial values in $Y(1,I)$, if an indication is given that a drastic change in the character of the solution has just been encountered and information from previous steps is not useful, a restart using the R-K method will provide a high order restart procedure, whereas formerly the program was restarted at first order. An algorithm was developed that works in this way.

In the original DIFSUB, if a step is repeated three times with different step sizes, and converges, yet the calculated error is still too large, the program is restarted at first order. This was replaced with the R-K restart. Also, if the program reduces to second order without a "significant" increase in step size, a R-K restart is initiated. Significant is defined

as being an increase in step size by a factor of RK, where RK = 10 initially but is reduced to 2, then 1 if this technique is unsuccessful in improving the step size.

This program was used on the step motor problem specified in the previous quarterly report. The results for comparison are

	<u>NFNS</u>	<u>EXEC TIME</u>
ORIG R-K	19,945	21.9
STIFF	12,366	38.0
ADAMS	5,080	17.6
R-K START		
ADAMS	3,599	13.95

Tests with other sample programs are planned to see if this is a significant improvement over the original DIFSUB starting procedure.

3.1.2 Sparse Eigen Problem (J. S. Deogun)

Linear Pole-Zero Analysis

Investigation of the sparse eigen problem was completed last quarter. Therefore, investigation of a new problem "linear pole-zero analysis" of electrical circuits was started this quarter. The problem of electrical network analysis is divided into four major areas:

1. linear DC and AC analysis
2. non-linear DC analysis
3. non-linear transient analysis
4. linear pole-zero analysis

Many survey reports in books and papers published over recent years have cataloged and compared most existing and proposed techniques for computer-aided network analysis in all four areas listed above. Our present concern is to study existing linear pole-zero network analysis techniques and programs and explore the possibility of developing a better technique. This quarter most of the time was spent in building a good background to start this

investigation. Beginning the introductory book NETWORK ANALYSIS by van Valkenberg, a few survey articles were scanned thoroughly to collect much of the available literature on the subject. An overview of the problem can be given as follows.

Under certain assumptions, network (transfer) functions take the form of quotients of polynomials in s having the general form

$$N(s) = \frac{p(s)}{q(s)} = \frac{a_0 s^n + a_1 s^{n-1} + \dots + a_{n-1} s + a_n}{b_0 s^m + b_1 s^{m-1} + \dots + b_{m-1} s + b_m}$$

which is a rational function of s (n and m being integers) and coefficients a and b are real and positive. $N(s)$, therefore, takes the form

$$N(s) = H \frac{(s-z_1)(s-z_2)\dots(s-z_n)}{(s-p_1)(s-p_2)\dots(s-p_m)}$$

Here $H = \frac{a_0}{b_0}$ is a constant known as scale factor and z_1, z_2, \dots, z_n and p_1, p_2, \dots, p_m are complex frequencies. When variable s takes the values z_1, z_2, \dots, z_n , the network function vanishes. Such complex frequencies are zeros of the network function. When s has values p_1, p_2, \dots, p_m , the network function becomes infinite. Such complex frequencies are the poles of the function. The network function is, therefore, completely specified by its poles and zeros and the scale factor. Poles and zeros of the transfer function can provide the same frequency response as the linear AC analysis. Even though, in general, more computational work is required for pole and zero analysis, it is still a good field of investigation as more insight is obtained by a knowledge of poles and zeros. Moreover, in many problems only the poles and zeros are of interest to the circuit designer.

3.1.3 Auto-Restart (B. van Melle)

The auto-restart facility for the numerical package was implemented this quarter and appears to be working satisfactorily.

Equation Parser

A new version of PARSE was written and installed this quarter. The new routine uses a left-to-right parse, replacing the old cumbersome top-down parse.

The operation of the new PARSE is fairly straightforward. The routine scans an equation from left to right, pushing operands and operators on a stack as it finds them. When operator precedence rules indicate that a previous operation can be performed, the stack is collapsed and the indicated operation is coded in the equation tree.

The relatively simple structure of the new program allows more efficient handling of the special functions. Previously, when SQRT or LOG was encountered, an extra equation was created in a temporary area using the argument to the function, and this equation was dealt with on a subsequent call to PARSE. Now it is possible to parse all the extra equations associated with an input equation during the same call. When SQRT or LOG is encountered, a new stack is built right on top of the old one, and the function's argument is parsed immediately. On reaching the end of the argument, the equation tree generated is moved to the output area, the extra stack is removed, and we resume where we left off. The argument can itself contain a special function, and the depth of such nested parses is limited only by the length of the equation (72 characters). The variable names created by PARSE have also been changed to allow easy recognition later on, so that they may be assigned initial bounds. For SQRT we use &SQRTnnn and for LOG, &DLOGnnn. Thus,

A = SQRT(B)
C = LOG(LOG(D))

becomes

A = &SQRT001
&SQRT001*&SQRT001 = B
C = &DLOG002
EXP(&DLOG002) = &DLOG003
EXP(&DLOG003) = D

A new operator, integration, has been introduced ('%' in EBCDIC).

It has precedence greater than + or - and less than * or /. It is implemented in roughly the same way as SQRT or LOG. For example,

E = F*G+H

becomes

E = &INTG004 + H
&INTG004' = F*G

Error diagnostics in the new PARSE have been made somewhat more helpful. In addition to describing the error, an error message includes the position of the equation scan pointer at the time the error was detected, thus pinpointing the problem.

3.1.4 Plot Package (W. Chung)

The plot package has been in successful operation throughout the quarter.

1. In addition to the main input of joystick hits, new forms of typed-in commands are included. The set of commands is not fixed but most of the joystick commands are acceptable. This allows user-defined variable names to be substituted for the internal variable names for easy recognition.

2. One more command <VAR NAME> is implemented for plot mode. This command followed by the joystick hit of listed var name gives the user supplementary information (internal name, user-defined name, position in the network).

3. List of commands for command mode

RECOVER; CLEAR; DISPLAY; PLOTMODE; RESTART; QUIT;
PRINT; EVALUATE; SAVE PIC; HARDCOPY; CORELATE;
SCALE; IDENTIFY; LIST; DELETE; FETCH; TUTOR

New commands will be explained briefly.

SAVE PIC--present picture on the screen is given a name by the user and saved in XFILE for the purpose of hardcopy production or later display.

HARDCOPY--the hardcopy output of the present picture or any picture saved in XFILE will be produced off-line. To do this one should follow the procedure of producing hardcopy using POT.

FETCH-- the picture in XFILE which was named and saved before will replace the present picture on the screen. Users type in the picture name.

TUTOR-- this command followed by any one of the above commands will give direction for the command to the novice user.

CORELATE--plot against var other than time will be given: independent and dependent variables are specified by the user. This will show the correlation between variables or parameters.

4. Smoothing of the plotted curve is controlled by the program.

Present method is based on the relative size of the minimum and maximum meshes. An improved algorithm is under study which will be based on the data selection algorithm. The data selection algorithm will select and save necessary data points which satisfy both a minimal number of points and a reasonable reproduction of smooth curve. The basic idea is to save more points when the relative rate of change of curvature is big. This will need further mathematical analysis.

5. The selection of a specific curve using joystick is determined by the minimum distance instead of vertical distance. This is important particularly when more than two curves cross each other with rather small

distances. Once the position of the hit is known, at most two data points are searched for each of which distance between the hit and the line segment of curve is computed using the geometry of triangles.

3.1.5 Item Analysis (J. Koch)

The handling of error messages was modified so that those messages that involve improper syntax in commands to ITEM are not put on ERRLIB.

The new equation parser was successfully attached to ITEM using the new parameter list that follows:

PARMLIST DECT		PARAMETER LIST		
ADDR	DS	A		PTR TO EQUATION TO PARSE
JCONS	DS	A		ADDRESS OF CONSTANT TABLE (CODE 6)
JELNM	DS	A		EI-NAMES (CODE 4)
JLOCL	DS	A		LOCAL NAMES (CODE 3)
JPARM	DS	A		PARAMETER NAMES (CODE 2)
JGLOB	DS	A		GLOBAL NAMES (CODE 1)
JLHS	DS	A		LEFT-HAND SIDE VARIABLES (CODE 2)
EXES	DS	A		ADDR OF ADDR OF SPACE TO OUTPUT EQU
LASTEQU	DS	A		ADDR OF LAST EQUATION OR ZERO
COUNT	DS	A		EQUATION # (TO BE UPDATED)
WORKX	DS	A		ADDRESS OF SB WORK AREA
ERRLIST	DS	A		ADDR OF ERRLIB ALIST
TRANS	DS	A		ADDR OF EBCDIC-ASCII TR TABLE
IFLAG	EQD	TRANS		FOR BYTE HAS INPUT/OUTPUT FLAG
LHSBIT	EQD	X'30'		ON IF LHS VARIABLE PRESENT
HALBIT	EQD	X'40'		ON IF PARSE HALTED
EXBIT	EQD	X'20'		ON IF PARSE FOUND FATAL EXCS
JAFILE1	DS	A		ADDRESS OF FILING SYSTEM

A semi-colon is placed at the end of each equation before PARSE is called.

More degenerate cases of networks were run to ensure that all types of networks that could possibly be constructed are accepted by ITEM.

MAP

The equation mapping routine that equates internal variables to user variable names was modified. Doubleword constants used to be printed in their internal (hexadecimal) form. To make these more readable by the user, routines were added to convert them to their base ten equivalents.

PDP-8

Design was begun on a language for the PDP-8 that will allow a graphics routine to be more easily programmed. I am learning the assembly language (PAL8) and the current system programs. Those routines that can be extracted, such as line drawing, will be included to minimize reprogramming. The new language will probably be interpretive to allow for address verification at run time and assembly of only the needed branches of the program. One type of language being considered is similar to one by Newman. It is state-diagram oriented with the program organized into blocks (states) with various actions possible from that state. Further evaluation of the language and its possibilities for implementation on the PDP-8 will be done during the next quarter.

3.1.6 FINDEX (J. Stynes)

Initial Conditions

This quarter a package was developed to allow the user to specify the initial conditions that are to be used when a network is analyzed.

1. Description

There are essentially two operations that might possibly be performed during the specification of initial conditions:

- a. setting a variable equal to a certain value
- b. setting limits on the range of values a certain variable may assume. (For example, a variable involved in a square root should be greater than or equal to zero.)

Before proceeding any further, it will be helpful to give a description of the types of internal variables. Every user variable is associated with an internal variable. (See 2nd Quarterly Progress Report 1972, pp. 63-71 for a more detailed description of the relation between user and system variables.) The type of internal variable that is assigned to a user variable depends on how the user variable appears in its equations. The different types of internal variables are listed below:

The user variable is assigned as a

- a. Y variable--if it is a non-linear variable--i.e., has non-constant coefficients. The Y variable is described by a two-dimensional array $Y(7,M)$ in which $Y(1,M)$ corresponds to the M-th Y variable, and $Y(2,M)$ corresponds to the derivative of that variable.
- b. YL variable--if it is linear--i.e., has constant coefficients.
- c. T variable--if it is a function of time (and possibly global variables).
- d. G variable--if it is a function of global variables only.

2. User's Guide

In most of the functions that follow it will be necessary to describe the variable that is to be initialized or bounded. This is done as follows:

(At this point, it is assumed that the reader has read the 2nd Quarterly Progress Report 1972, pp. 63-71, which describes the method of uniquely defining a variable.)

To define the variable, V, one would write:

V(<top level element>:<corresponding PDPNO>,
<next level element>:<PDPNO>,..., <bottom level element>:<PDPNO>)

So, for example, to define the variable U, which is on node 1 in the element STAR, which is in the network 3BODY, one possible description would be

U(3BODY,STAR:0)

Note: If both element and PDPNO are not specified, the colon is dropped.

Some examples of variable descriptions follow:

U(0)

V(ELEMENT:3,4)

X(ELEM1:2,ELEM2,3)

In the following discussions, <variable> will be used to represent a variable description.

a. Setting a variable to a certain value

It is possible to set any variable to an initial value, but, in general, only Y variables are initialized. There are two ways to assign a value to a Y variable

1) GVN <at least one blank><variable> = value where value is a number. The effect of this statement is to set the variable to the specified value, and to set an indicator which indicates that if the variable/derivative is given a value, the derivative/variable will be solved for. Thus, if the GVN function is used to give a value to the variable, V, V' will be assumed to be unknown.

2) If it is desired to set a variable to a value, without changing the indicator, it is simply necessary to omit the GVN in the assignment statement.

Some sample assignments are

$$\text{GVN } \underline{\quad} \text{U(STAR)} = 3$$

This sets the internal variable corresponding to U(STAR) equal to three, and sets the indicator so that U'(STAR) will be solved for.

$$\text{U(STAR)} = 4$$

This only sets the internal variable equal to four. The indicator remains unchanged.

Note: At the start of initial conditions, the Y variables, Y(1,M), are set to a random value between one and two, their derivatives, Y(2,M), are set to zero, and their indicators are set to solve for Y(1,M).

3) Aside from the internal variables described above, there are also variables which are used as integration parameters. These are

TSTART--contains the initial value of time

TEND --contains the final value of time

DEL --contains the size of the error control
for the initial value program (DIFMF3)

EPS --contains the size of the error control
for the integration step (DIFSUB)

These may be given a value by entering

parameter name = name

For example, TSTART = 0

EPS = .1

To list the values of these integration parameters, type LP.

b. Setting bounds on a variable

Only Y variables are permitted to have bounds. The user may enter upper bound, lower bound or both. The formats for entering bounds are as follows:

- 1) To set a lower bound:
value< <variable>
- 2) To set an upper bound:
<variable> <value
- 3) To set both bounds:
value< <variable> <value
where value is a number

It is also possible to delete bounds by using the following functions.

- 1) To delete lower bound:
DELL <any # of blanks> <variable>
- 2) To delete upper bound:
DELU <any # of blanks> <variable>
- 3) To delete all bounds
DELA <any # of blanks> <variable>

3. Implementation

There are basically three subroutines which comprise the initial conditions package: INICON, GETVAR, FINDEX. INICON is the main subroutine which performs the requested functions and assignments. GETVAR scans a user text line for the function and variable, or assignment statement, while FINDEX finds the internal variable associated with a given user variable. Thus, INICON calls GETVAR to scan the entered text line, and GETVAR calls FINDEX to get the internal variable. The internal variable and the function

are then passed back to INICON for processing. FINDEX was described in the previous quarterly. GETVAR and INICON are described below.

a. INICON

The calling sequence for INICON is:

CALL INICON(Y,IND,NY,YL,NL,LIST,PLIM,MM,TSTART,TEND,G,NG,DEL,EPS,MAX)

where

Y --(Real*8) is a 7 x NY array of dependent variables and their derivatives. Y(1,J) is the variable, Y(2,J) is its derivative.

YL --(Real*8) is a vector (length NL) of dependent linear variables. (NY+NL = Number of equations in system (usually))

IND --(Integer*4) is a vector (length NY). Each component corresponds to one of the Y's.
If IND(J) = 1, then Y(1,J) is unknown, and Y(2,J) is given.
If IND(J) = 2, then Y(1,J) is given and Y(2,J) is unknown.

LIST --(Integer*2) is a 2 x MM array containing information about bounds
LIST(1,K) = J where

J=1 means lower bound on Y variable given.

J=2 means upper bound on Y variable given.

LIST(2,K) is the index to the Y array in terms of bytes past the base address. For example,
Y(1,1) has index 0, Y(1,2) has index 56. In general, Y(I,J) has index: $8*(I-1+7*(J-1))$

PLIM --(Real*4) a vector (length MM) containing the bounds.

PLIM(K) contains the bound described by LIST(1,K) and LIST(2,K)

If both bounds are specified, the lower bound goes in PLIM(K)

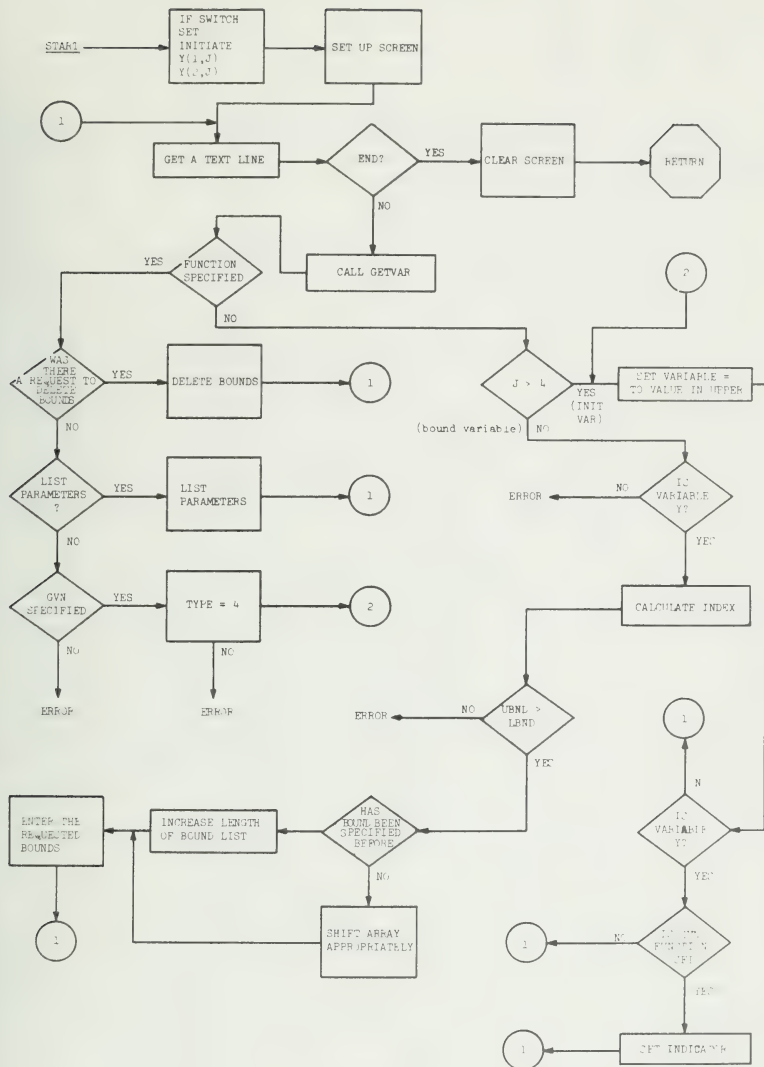
and the upper in PLIM(K+1)

LIST(1,K)=3, and LIST(1,K+1) is ignored. Thus, if
LIST(1,K) = 1 or 2, it is assumed only one bound
has been stored.

TSTART,TEND,DEL,EPS--(Real*8) are the previously described
integration parameters.

MAX --(Integer*4) is the maximum size of the Y arrays

On the following page a **general** flowchart is given.



INIS-AN

b. GETVAR

GETVAR is a general purpose scanning program. The general form of the type of line it will accept is

<Function><at least 1 blank><expression>

where <expression> is either an assignment statement, or a specification of bounds (of the type described above).

The calling sequence for GETVAR is

CALL GETVAR(TEXT, LENG, TYPE, INDEX, INDIC, FCN, UBND, LBND)

where

Input: TEXT--contains the line to be scanned

LENG--contains the length of the line

Output: TYPE--is the type of variable (see FINDEX for variable types)

INDEX--is the index into the variable array

INDIC-- -1 Error in input

1 Variable has lower bound specified

2 Variable has upper bound specified

3 Variable has both bounds specified

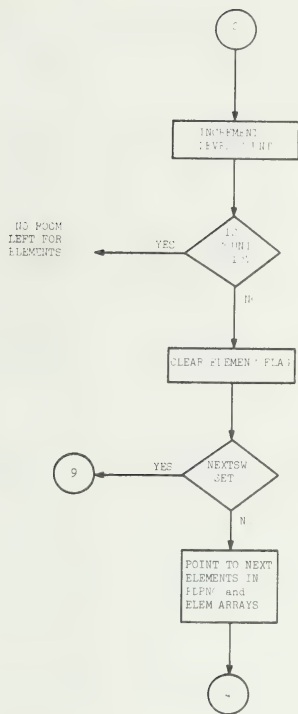
4 Initial value specified

FCN --contains function requested. If no function was specified, its value is 0

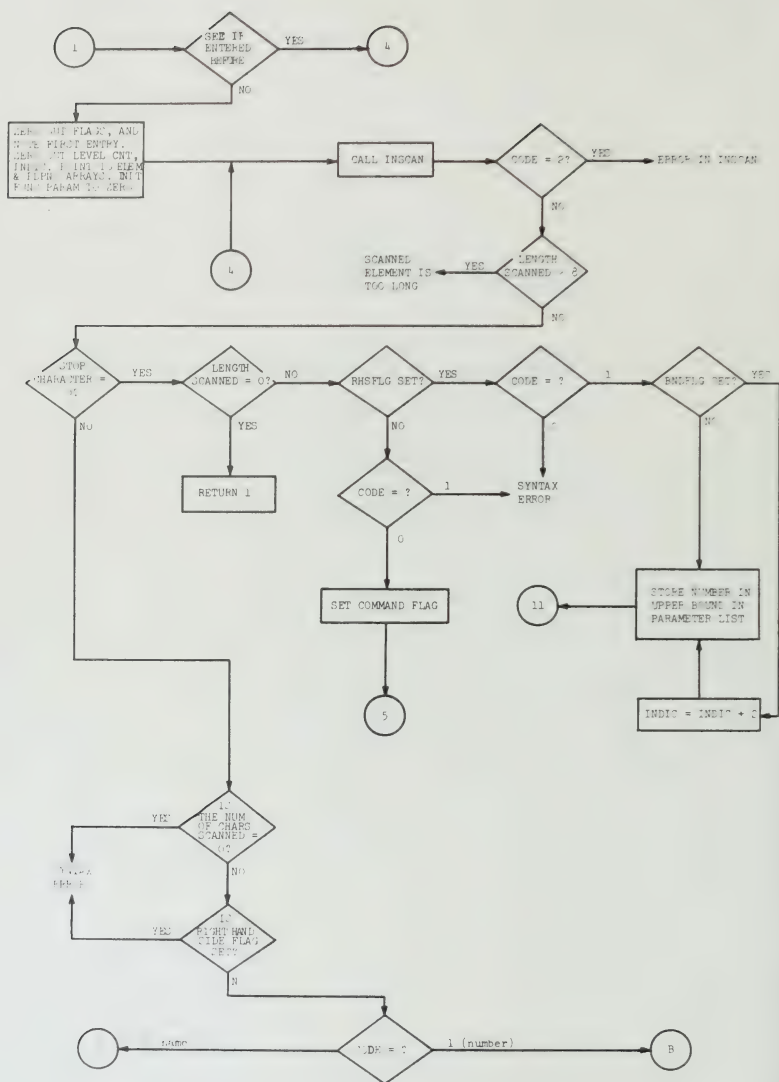
UBND--contains the upper bound if given, or the initial value if INDIC = 4

LBND--contains the lower bound if given

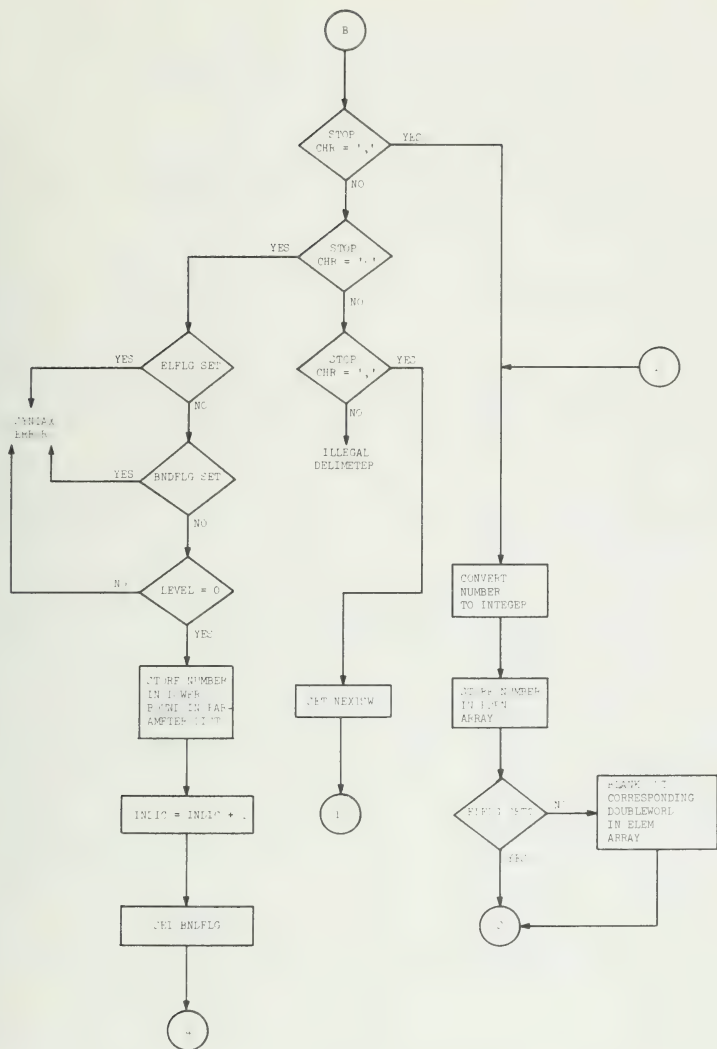
On the following pages is a fairly detailed flowchart for GETVAR. There is one confusing complication that should be mentioned. It is possible that the user's variable is the negative of one of the Y's or YL's (one of the parameters returned by FINDEX will show this). In this case, the initial value or bound must be negated, and lower bounds become upper bounds and vice versa.



GENVAP

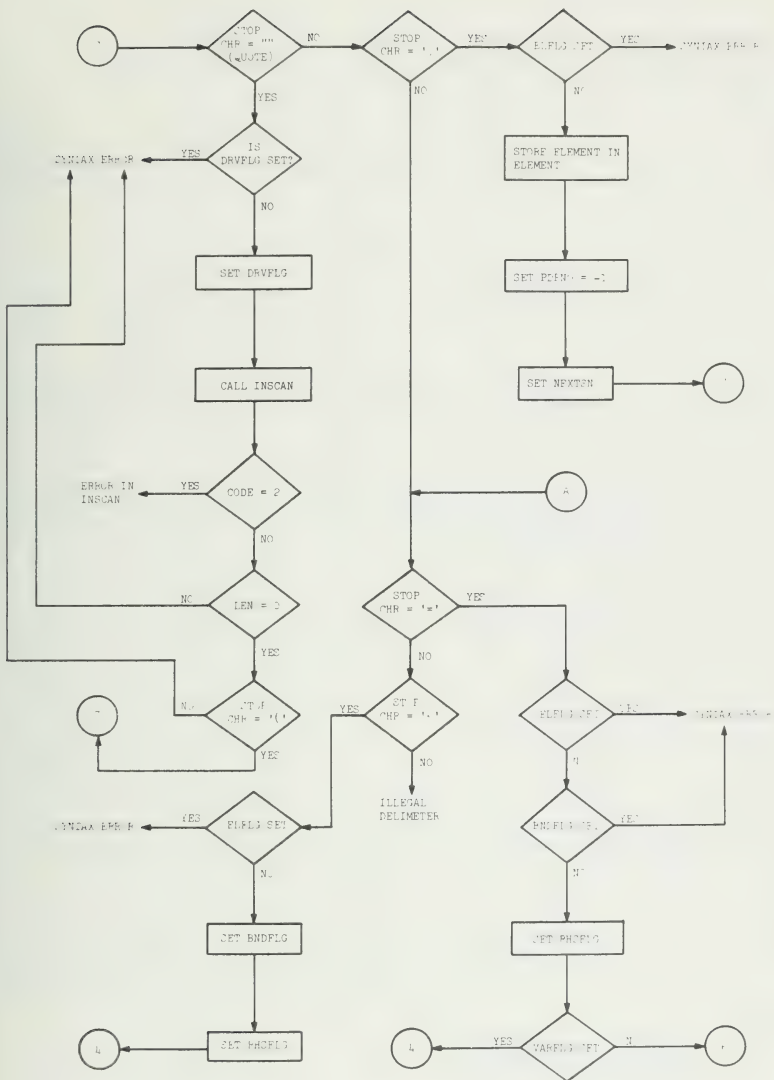


REINIT

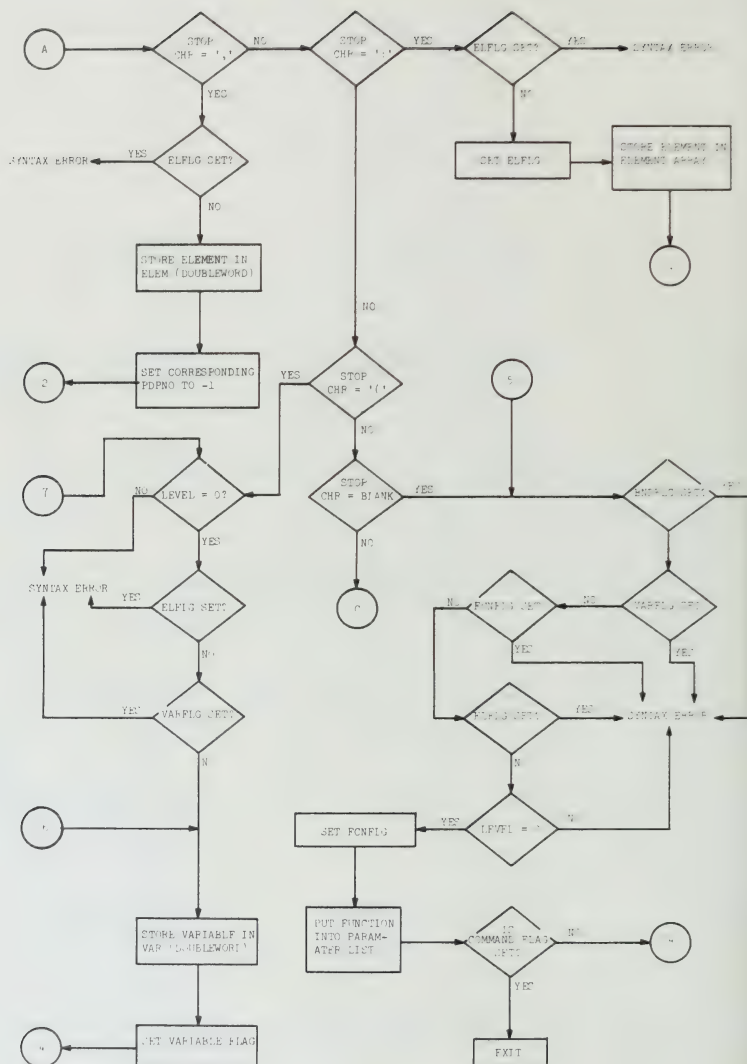


END OF





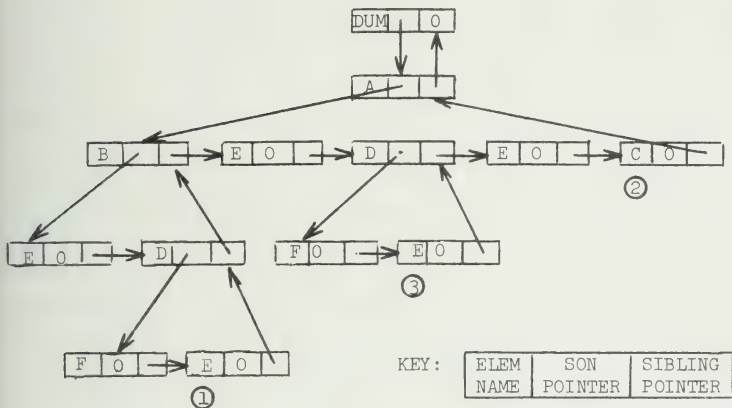
GETVAR



GETVAR

The following diagram appeared in the 2nd Quarterly Progress

Report 1972 on page 69. Please note that it has been corrected.



3.2 Illinois Graphics Computing System (IGCS)

3.2.1 Overlay Supervisor (D. Mueller)

The overlay supervisor was completed during the Spring of 1972. This package brings predefined modules into core and provides facilities for modules to overlay themselves. Overlays may be either on disk or DECTape, and facilities are provided for updating the supervisor tables should the location, length or starting address of an overlay change. Two facilities are provided for initializing the monitor tables. One allocates dynamically and is intended for debugging purposes. The second has the tables predefined and is intended for production use. Programs are also provided to create overlays from load modules produced by the Linker.

The supervisor was revised during the Summer to make debugging easier. A user's manual is in the process of being written.

3.2.2 A Drafting Language (ADL) (T. Runge)

ADL is a general purpose drafting language which facilitates the communication of graphical constructions and annotations. The general strategy devised for implementing ADL via four overlays has been described previously.

TABLE 1
ADL Instruction Labels

<u>Instruction</u>	<u>Letter Code</u>	<u>Geometric Element Options</u>	<u>Line Type Options</u>	<u>Coordinate Options</u>	<u>Referencing Options</u>
straight line	L	--*	H <u>n</u> or C	P	N
circle	C	A <u>n</u>	H <u>n</u> or C	P	N
ellipse	E	A <u>n</u>	H <u>n</u> or C	P	N
center line	Q	U	--	P	N
text string	W	V	--	P	N
marker	M	--	--	P	N
plane entity	D	--	--	--	--
entity definition	F	--	--	--	--
next image	I	--	--	P	N
translate origin	O	--	--	--	N is assumed
end terminal point	T	--	--	P	N
end scale	S	--	--	--	--
comment	/	--	--	--	--

*Dashes indicate that a particular type of option is not applicable to a particular instruction.

Half center lines, consisting of one short (usually centered) dash and one long dash, rather than two, can be specified by choosing the U option to the special center line geometric element. The default option is one short and two long dashes.

Vertically oriented text strings can be specified by choosing the V option. The default option is horizontal orientation.

Line Structure Options:

These options, available to the geometric elements, specify what type of line structure other than solid is to be used in plotting the element. Hidden lines--consisting of dashes separated by spaces--can be specified by choosing the H^n option. The digit n specifies the endpoint conditions. Centerlines (other than the special centerline element) consisting of alternating long and short dashes separated by spaces can be specified by choosing the C option. Default for all elements is a solid line--entirely plotted. The method used to control plotting under these options is general enough so that other plotting options can easily be added if desired.

Coordinate Options:

These options, available to all instructions which include coordinates except the translate origin instruction, tell the compiler what type of coordinates are being used throughout the entire instruction. Polar coordinates can be specified by choosing the P option. Default is rectilinear coordinates.

Referencing Options:

These options, available to all instructions which include coordinates except the translate origin instruction, tell the compiler the reference point to which all coordinates in the instruction refer. Non-incremental referencing--all coordinates referring to the current plot origin--can be specified by choosing the N option. Default is incremental referencing. In it, all coordinates refer to the current terminal point, a reference point maintained by the compiler which is usually associated with a just-completed element. Geometric elements under incremental

referencing make further use of the terminal point. For these, the current terminal point is assumed to be one of the line or arc endpoints and thus the endpoint need not be specified. After compiling the figure, the terminal point is then set to the other line or arc endpoint. Thus, a sequence of connecting lines and arcs can be specified in a very straightforward manner using incremental referencing.

Procedures:

The compiler provides for four marked points. The user can 'remember' the coordinates of a point by marking it with a mark instruction. He can later 'recall' those coordinates by substituting M_n for a coordinate pair in an instruction. The digit n refers to the particular one of the four marked points that is being 'recalled'. Coordinates specified in an instruction by means of marked point recollection are not subject to the referencing and coordinate options in effect. They are assumed to be referenced non-incrementally and in rectilinear coordinates.

An image creation-insertion facility is provided. A display entity can be defined by placing the instructions which describe it between a define entity and an end definition instruction. Then, by means of an insert instruction, its image can be inserted with all coordinates of the entity translated by an offset specified in the insert instruction. The entity's image can be inserted in as many different places on the paper as desired, but the image is deleted from memory by any ensuing define instructions.

The plot origin, initially set at the lower left corner of the paper, can be translated at any time by means of the translate origin instruction. The offset is specified by the user. When the plot origin is translated, all marked points and the terminal point undergo an identical offset.

The terminal point, initially set at the lower left corner of the paper, can be reset at any time by means of the set terminal point instruction. The desired offset from its current position (incremental referencing) or from the plot origin (non-incremental referencing) is specified by the user.

The scaling factor, initially set to one, can be changed at any time by means of the rescale instruction. The desired rescaling factor is specified by the user.

A user's guide, containing a more detailed description of instructions and options, is currently being written.

3.2.3 Plot Sectorization Overlay (T. Runge)

Work on this overlay, begun in the previous report period, was continued. Briefly, geometric elements and text string specifications that have been compiled are sorted into disjoint sets or sectors, according to the interval into which the first point of the element or string to be plotted falls. Parameters for the iterative plotting in each sector are also established. A more complete description of the algorithm is contained in publication 1 "Illinois Graphics Computing System," Departments of General Engineering and Computer Science, University of Illinois, Urbana, Illinois.

Minor alterations necessitated by instruction changes in the compiler are yet to be incorporated. Full scale debugging will be possible upon completion of the third overlay.

3.2.4 ILLISM-E (W. Tam)

The ILLISM-E compiler is ready for merging with DIFSUB. Only limited output capability is available as the interface with the proper output packages is still under development. The ILLISM-E compiler generates executable (absolute) codes directly in core. An attempt was made to allow the user to specify arithmetic routines in element definition statements. This posed considerable problems as the executable code is generated during run time and most of the arithmetic routines are in the library. The tentative solution is by keeping a table of the arithmetic routines in core. Investigation is underway to find the possibility of keeping the routines on disk and calling them in when required.

Modifications were made in the diagnostic handling procedures so that the compiler can also be used for batch mode processing. The current version (teletype interactive) types the error messages onto the teletype as soon as an error is detected in the input statements. In the batch version, all error messages are stored away in a data file, which can be directed to the line printer when compilation is complete.

3.3 Graphical Remote Access Support System

3.3.1 OS/8 Operating System (J. Nickolls, C. Segre)

A device handler which uses the 2701 channel as a line printer has been written and successfully implemented along with an IBM 360/75 monitor program which receives buffers from the PDP-8 and prints them on the 1403 line printer. This remote printing facility is most helpful when it is impossible to use the Inktronic printer.

All source code for GRASS has been converted to OS/8 format using CONVERT. It is now possible to use the disk as the OS/8 system device by running SWAP.

HELP, a program which allows examination and modification of the disk and the tapes is now operational, it is helpful for patching both OS/8 and GLOAD tapes.

3.3.2 Information Retrieval (J. Nickolls, C. Segre)

IR has been slightly modified and is currently being debugged along with a new version of GRASS.

3.3.3 Monitors (J. Nickolls, C. Segre)

GLOAD/GEXIT

More options and keyboard commands have been added, the procedure for loading GRASS from tape is now

1. Start up ACID or MON8/I in the PDP8/I
2. Mount load tape on unit 8 and library tape on unit 4
3. Bring up DECTape system and type 'GLOAD'
4. Type in option letters in response to 'OPTIONS:'

followed by carriage return. The option letters and their meanings are

T--restore local library from tape as of last GEXIT

U--add specified user's files to the local
library from the tape

D--use current local library on disk

N--initialize clean library with no files

R--reload resident programs from disk

S--load disk and core with all system programs from tape

A--load PDP8/I with ACID

if no options are specified in one or both groups, the default options are 'U' and 'R'

The procedure for exiting the library onto tape is now

1. Bring up DECTape system ('DT' command on console)
2. Put write enabled on unit 4 (library tape)
3. Type 'GEXIT'

NOTE: All users must be logged out at this point in order to exit correctly.

4. Type in option letters in response to 'OPTIONS:' followed by a carriage return

The option letters and their meanings are

D--write all disk files to the tape

U--write specified user's files to the tape

in case of no options being typed, the default option is 'D'.

In both GLOAD and GEXIT, if the 'U' option is specified, an additional question ('USER NAMES:') will be asked. Respond with problem specification numbers and user names (ex: '0072POOHBEAR'), with the names separated by commas. End the string of names with a carriage return. If more than one line of names is needed, end the string with a comma and carriage return, and continue on next line.

Program Segments

GUT1

To facilitate text editing, GUT1 has been modified to allow editing in various text areas. Formerly, it was impossible to edit in more than one area without exiting and re-entering the text editor. Currently, if the screen displays the message 'REORG BUTTON ENABLED', it is possible to hit any text area with the reorg button and subsequently edit it.

The display has been slightly modified due to the implementation of the function buttons. The menu scrolling functions, formerly 'P' and 'N' in the light button area, are now 'PREVIOUS' and 'NEXT' in the answer area. The 'REORG' function is now a button on the joystick box.

Several light button functions have been implemented. The '<AV>' functions will now produce appropriate arrowheads at the last draw area hit. The 'X' function on the lower left side will display all terminal connections when hit.

GLASP (PDP-8 Local Monitor)

A new system with major revisions is now being tested, this new system will contain joystick pushbutton functions and keyboard functions from the operator's console. Also in this system are better menu and library routines and many minor changes to aid the GRASS user.

3.3.4 GLOM: New Filing System (A. Whaley)

A new filing system is being written for the grass system.

Reasons for this work are as follows:

1. Old system (XFILE) has fixed limit on number of file names allowed.
2. XFILE has too simple a method of disk allocation that causes severe problems when space on the disk is exhausted.
3. When free space on the disk is lost due to repeated crashes of user programs under test, there is no simple way to regain this space.

The new system (GLOM) is a combination of the BLAM¹ filing system and XFILE, the old graphics filing system currently in use. The main features of GLOM are as follows.

Data Structure

The tree structure of BLAM and much of the associated programs in BLAM for manipulating the data structure have been borrowed for use in GLOM. The tree structure is already described elsewhere, so only modifications will be discussed here. The major change was caused by allowing individual records to be as long as 65,535 bytes. If a record is shorter than an internal fudge factor (currently set at .75 of the fixed size of blocks used), it is stored in a data block as described for BLAM. If, however, the record is longer than this, it is broken up into pieces each of which is set to the fixed length of the blocks used by GLOM.

Channel Programs

When large data blocks are read or written, all I/O requests for the blocks involved are entered before the actual channel program is formed. Requests are made by internal subroutine BUILDCCW. Channel programs are constructed in the most optimum fashion possible and executed when EXECUTE is called. All requests in a single cylinder for which space is available are handled by a single channel program. Rotations are minimized by attempting to do consecutive records at all times. Use is made of the head seek command in order to accomplish the optimization. During

¹ "A Failure Tolerant Filing System," Department of Computer Science Report UIUCDCS-R-72-506, University of Illinois, Urbana, Illinois 61801.

initialization, GLOM modifies the OS Data Extent Block in order to permit itself to do seeks of this type. The control block is in protected storage.

Storage Allocation

Storage allocation on the disk is a completely new facility not found in BLAM or XFILE, and which allows relative addressing of disk blocks and multiple extent and multiple volume OS data sets. In addition, provision is also made for adding additional extents should space become exhausted via the secondary storage allocation parameter. Each group of extents on a volume is accessed through a DCB assigned to that volume. Only one DD card is used for specifying all disk space.

Re-evaluation of Disk Space

When initializing the filing system, one may request the following options:

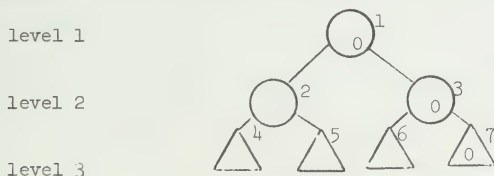
1. Reformat entire data set, then do (2). This causes fixed length blocks containing zeros to be written on all tracks of the data set.
2. Initialization--data structure is set to a single top empty catalog file. All storage allocation and control bit maps are located by a new DIB block and are initialized.
3. Re-evaluate--storage allocation is recomputed by freeing all blocks and then allocating each block found in the data structure. Re-evaluate occurs automatically if not completed by the previous attempt. A bit in the DIB indicates that re-evaluation is not yet completed.

Full use is made of the disk optimization facilities for these operations.

Record Location

The current version of BLAM used in all file maintenance with the local timesharing system is optimized for record location. When a record is first accessed in a file, a PATH is set up in the file control block (FCB in BLAM, XDCB in GLOM), to indicate the vertical path from the top of the tree leading to the located record. Each entry contains a disk address and a displacement within the block at that address to the record accessed. All but the last record would be directory entries describing blocks lower in the tree. The last record described by PATH would be the data record described, or in the case of very long records, a description of where each piece of the long record was located. For long records, the actual data is not considered part of the tree structure.

Optimization uses this path to locate the next record requested by going directly to the previously accessed record and if the record is focused to be in that block, going forward or backward from the previous record to either locate the requested record or the place where it should be. All records in a file are in order by their 8 byte keys (line numbers) at all times. Clearly for sequential reading of a file, this method will be quite rapid. For inserting lines into an empty file sequentially, this method fails. Imagine a file under construction:



The records are being added in a sequential manner. PATH indicates the last record added. When the next record is to be inserted, BLAM first does a locate (LOCREC) on its name in order to find the point in the file where the

record should go. INSERT then uses PATH to install the record. Optimization in LOCREC first goes directly to block 7 and determines that the record is not in this block. LOCREC then goes to block 3 to see if more data blocks exist after block 7, then to block 1 to see if more directory blocks exist after block 3. Then not being exactly sure how it arrived at block 1, LOCREC goes into its normal unoptimized section and works back down from block 1 constructing a PATH almost identical to the one already there, the difference being in the last entry, which indicates the EOR (end of record marker) right after the last record added.

Obviously, attempts at optimization must fail for some strategies. This is the sort of inefficiency one would expect from repeated accesses to widely separated parts of a file. Sequential additions should not be subject to these problems, however, and so a flag (XDCEND) was created in GLOM which indicates that PATH is set to the last record in the last block at each level except the bottom-most level. This would infer positioning at the last block in the lowest level. Optimization uses this flag to avoid going back up the tree.

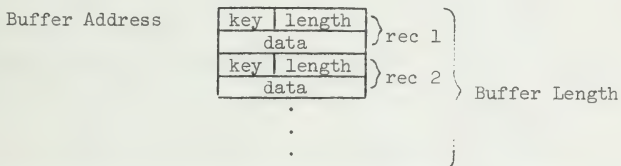
A more general solution to this problem would be a flag at each level in PATH to prevent this inefficiency when adding to the end of part of a file. This type of usage is uncommon, however, and not worth the added complexity and inefficiency. For the same reason a beginning of file position flag was not added for the case of adding records to the front of a file.

Command Expansion

BLAM already has a fairly full set of commands as it is used in several applications. Some new commands have been added to GLOM, however, as a result of some new design features. Facilities for reading forward and backward have been added, implemented through backward pointers in each record. The following read operations are now available:

XREAD read this or next if named record not found
XREADN read next in sequence from where PATH is set
XREADB same as XREAD but goes backwards
XREADNB same as XREADN but works backwards

A full set of commands also exist for reading in each mode which provide only the key and length of data for the record. As in XFILE, GLOM allows a record count for each read or write command. After each command, additional sequential commands in the indicated direction take place until the count goes to zero. Record format for reading and writing is the same as for XFILE:



The exact format for many commands has not been worked out, but a full list will appear in the next quarter.

3.4 Computer Maintenance and Construction

3.4.1 Graphics-8 Hardware (C. E. Carter)

Data Disk

The disk is back in-house after another trip to the factory. As soon as all other activities settle, it will be checked on-line to determine its usefulness.

Systems Industries Disk and Controller

Satisfactory operation of this new disk and controller over the past six months has given us confidence that it will continue.

On-Going Projects

1. Work continues of the PDP-8 core memory extension.
2. Investigation on modem interfaces for dial-up terminals has started.
3. Ordering of a hard copy unit was completed. It will be used with any one of the existing terminals.
4. The Computek terminals will be examined for interfacing of graphic tablets.
5. Planning for terminal #4 has begun.
6. The second joystick is in operation and proper hardware has been located for additional ones if required.

338 Display Address Counter

BIT 10 always 1--address bit driver hung on. Replaced.

PDP-8

1. Would not run any peripheral equipment. BAC five driver shorted. Replaced.
2. Switches two and ten not always making contact. Replaced.

Comptek

1. Put bad on joystick box on terminal #1. Replaced.
2. Multiplexor nut working correctly on terminal #1.
Bad board contacts. Cleaned.

Inktronic

Bad LF motor and clutch assembly. Replaced.

3.4.3 FDP-7 and 630 Maintenance (E. Pelg)

1. Replaced lamps in bits 1 and 12 AC register.
Replaced lamps Read Enable Indicator.
2. Channel interface would not read back correctly from 360, bad R121 location 3H28.
3. R141 bad in channel interface.
4. FDP-7 down for a week due to air conditioning maintenance.
 - a. Scanner test would not run, found bad PCB in interrupt logic M14-R141 PCB.

- b. Bad PCB in decoding circuit PDP-7, R121 location 3H25.
 - c. Channel interface erring both data and address transfers in write cycle, bad multiplexor PCB R141 location 3220, also R141 location 3D20 address register was intermitten.
- 5. Replaced data line #15 to port 76, data line #1 back to port 50.
 - 6. Channel interface reading back incorrectly, bit 11 MA R141 location 3C26.

PUBLICATIONS

Gear, C. W. and Tu, K. W. "The Effect of Variable Mesh Size on the Stability of Multistep Methods," Submitted to SIAM Journal on Numerical Analysis.

Tu, K. W. "Stability and Convergence of General Multistep and Multivalued Methods with Variable Step Size," Department of Computer Science Report No. 526, University of Illinois, Urbana, Illinois. Submitted in partial fulfillment of the requirements of the Graduate College for the degree of Doctor of Philosophy in Mathematics.

U.S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

C00-1469-0213

2. TITLE

3rd Quarterly Progress Report 1972

3. TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

- ☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

C. William Gear
Professor and Principal Investigator

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature

Charles W. Gear

Date

September 30, 1972

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION
RECOMMENDATION:

8. PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

4. CENTER FOR ADVANCED COMPUTATION

Report Summary

This document reports progress on ARPA contract DAHCO4-72-C-0001 entitled "ILLIAC IV Applications Research at the Center for Advanced Computation, University of Illinois at Urbana-Champaign." The principal objective of this program is the development and testing of numerical techniques and software systems for use of ILLIAC IV over the ARPA Network. This is being accomplished through activities in the following areas:

1. Development of numerical techniques suitable for parallel processing in the areas of:
 - a) Linear systems of equations
 - b) Algebraic eigenvalue problems
 - c) Linear programming
 - d) Graph algorithms
 - e) Approximation of functions
 - f) Determination of real roots of polynomials
2. Development of an ILLIAC IV multispectral image processing system
3. Development of ILLIAC IV language for the phase II system
4. Development of programs in large scale calculations such as input-output economic modeling, quadratic assignment algorithms for various classes of spacial allocation problems, and atmospheric dynamics.

In addition, education of segments of the ILLIAC IV user community was accomplished through seminars, classes, and the development and dissemination of tutorial materials.

4.1.1 Introduction

During this period the Center's B6700 computer was disconnected (June 30th) and at least three weeks elapsed before we had reliable access to the simulator on the B6700 at UCSD. This has resulted in some delay in completing some of the ILLIAC IV codes.

4.1.2 Computational Methods in Linear Algebra

4.1.2.1 Solution of Systems of Linear Equations

The conjugate gradient method for solving the linear system of equations $Ax = b$ [1], has been coded in GLYPNIR and fully debugged, and a document is forthcoming. This algorithm is especially designed for large sparse matrices A with regular structure. It requires as an input a routine to compute Ay and $A^t y$ for a given y rather than storing the matrix A . To this end a high level language for matrix specification is being developed together with assembly language routines for handling such sparse matrices on the ILLIAC IV.

4.1.2.2 The Algebraic Eigenvalue Problem

A study of an algorithm for finding the eigenvalues of tri-diagonal matrices on the ILLIAC IV using a generalization of the bisection method [1, 2] has been completed. The results compared favorably with the QL algorithm [1] on serial machines. The results of this study will be published soon. Work has also continued on finding suitable parallel algorithms for finding the eigenvalues and eigenvectors of large sparse matrices that arise from the numerical solution of elliptic differential equations (using the finite element methods for

example). A comparison between Householder's method [1, 2] and Lanczos' method [2 - 4] to reduce such matrices to the tridiagonal form on the ILLIAC IV is almost completed.

We also obtained the eigensystem subroutine package (Eispack) from the NATS project at Argonne National Laboratories and had it installed on the University of Illinois' IBM 360/75 to provide an excellent set of routines that would enhance the development and testing of new algorithms. We are now in the process of adding this package to the library of the SPEAKEASY system [5] on the UCLA 360/91 for interactive usage. Hopefully this will also be used by the ARPA network community.

4.1.3 Linear Programming

The revised simplex code, described briefly in the previous semi-annual report, is largely written; however, it has not yet been debugged en masse (or with large data bases). This is mainly due to lack of complete documentation of the ILLIAC IV I/O. Since this I/O dependence will, most probably, be a stumbling block during the early months of ILLIAC IV availability, it was decided to consider instead for the next six months a new algorithm for linear programming developed by Gill and Murray [6] and Saunders [7]. This replaces the usual product form of the inverse of the basic matrix by a factorization into a lower triangular and an orthogonal matrix, with only the lower triangular matrix being stored. This method seems superior in both numerical accuracy and the greater sparseness of the lower triangular matrix compared with the product form of the inverse. This algorithm has already been implemented in ALGOL on the UCSD B6700, and is being implemented in GLYPNIR for the ILLIAC IV. The package is being coded

so as to need little or no I/O and might therefore be executable under the earliest operating system. If the sparse matrix is of a given structure, the implicit generation of Ay and $A^t y$ mentioned in (4.1.2.1) could greatly enhance the number of rows that can be handled in core.

4.1.4 Graph Algorithms

The ALGOL and ASK programs for reducing a matrix A to the block triangular form $B = PAP^t$ where P is a permutation matrix, were modified in several ways thus delaying documentation of the algorithm. The ILLIAC IV performance exceeded our wildest dreams. An operation count on the ASK code for Warshall's algorithm [8] for an $n \times n$ matrix showed that for $n < 2000$ the ILLIAC is 29,000 times as fast as a B5500 on the same problem. The largest boolean matrix which will fit in core requires only eight seconds of processor time. This ASK program is many times faster than the one mentioned in the last semi-annual report. A report is forthcoming.

We also considered the problem of clique detection in graphs. A highly parallel algorithm was developed and was compared favorably with two algorithms tested at the University of Toronto [9]. An ASK program is partially written for a 64 node graph. Clique detection has applications in information retrieval.

4.1.5 Approximation of Functions

A recent effort has been started in the area of uniform approximation of vector-valued functions. An algorithm has been developed [10] and is being implemented. This will be followed by implementation of an algorithm for simultaneous best approximation of

sets of experimentally-determined exponential decay curves. Results of the latter will be incorporated into a revised version of [11].

Both of these algorithms are based on the solution of linear inequalities to obtain improved approximations, since a Remez-type algorithm does not seem feasible. In the case of single-function linear approximation, the inequality approach, although applicable, has been studied very little and probably never automated. If the tests for simultaneous approximation work out well, it may be worthwhile to examine the method as an alternative to the Remez algorithms.

The applicability of the inequality approach to various types of single-function nonlinear approximation will also be investigated. For example, curve fitting by sums of exponentials (a program of considerable importance in analyzing chemical reaction data) presents serious difficulties when carried out by a Remez-type method [12].

Finally, a survey will be made of algorithms currently available for the solution of integral equations. An effort will be made to determine which algorithms are "best" (depending upon the particular problem to be solved), as well as to identify ways in which existing algorithms may be improved and problems for which new algorithms must be constructed.

4.1.6 Finding Real Roots of Polynomials

An ASK program has been fully debugged and tested for finding real roots of polynomials with real coefficients using a generalized bisection method. A document is forthcoming.

4.2 ILLIAC IV MULTISPECTRAL IMAGE PROCESSING

4.2.1 Introduction

In support of the earth resources observation and monitoring objectives of the ERTS/EROS programs of USGS/DI and NASA, the Center, in collaboration with the Laboratory for Applications of Remote Sensing (LARS) of Purdue University, has proposed to design, implement, and assist in the evaluation of an advanced multispectral digital imagery processing system for automatic interpretation of high-altitude aircraft- and satellite-gathered data and detection of land-use and resource boundary changes. Such a system would also be applicable to problems in aerial reconnaissance.

4.2.2 Research Findings

CAC investigations during the last six months, conducted in collaboration with LARS which is supported by NASA, have led to the following conclusions.

First, the ILLIAC IV should be quite efficient in executing the large scale calculations required for digital processing of multispectral images. An early examination of input/output data transmission rates with respect to central core processing speeds has shown that the full capacity of the ILLIAC IV could be exploited by such an image processing system.

Second, the archival laser store associated with the ILLIAC IV would seem an adequate device for storage and retrieval of the large quantities of data associated with multispectral images and the interpreted resource information processed from these images.

Third, an appropriate portion of the software package LARSYS, developed at LARS for research in remote sensing, could easily be implemented in parallel fashion on the ILLIAC IV. Together with CAC-developed

data and information management software, this would provide immediately a powerful capability for multispectral image interpretation and a sound foundation for future ILLIAC IV image processing and pattern recognition experiments.

Fourth, the ARPA Network seems a practical means for interfacing numerous, geographically-dispersed national management and planning agencies. ILLIAC IV image processing and pattern recognition research by other members of the ARPA community should also be facilitated.

4.2.3 Proposed ILLIAC IV Implementation

As presently proposed to USGS/DI and NASA, the system would include, at a minimum, parallel software for raw data management, multispectral cluster analysis, classification of image elements into aggregate categories, digital registration of multiple images, automatic change detection capabilities, and basic information management services necessary for tabular report generation and automated mapping procedures. Also included would be the development of remote-inquiry, serial software to allow decentralized access to the ILLIAC IV processing system and peripheral storage devices.

Through this project (and in accordance with the objectives and spirit of the ARPA Network), discussion has resulted between CAC and the Image Processing Laboratory of USC concerning picture-processing systems sharing. CAC has discussed with NBS possible connections of D.C. area Federal agencies to the ARPA Network via the NBS ANTS facility. Communication with NASA-Ames ILLIAC IV users has been increased.

A preliminary proposal was submitted to USGS/DI and NASA by CAC and LAMP in July. Contractual arrangements are still being negotiated. The Center is hopeful that some FY-73 support from USGS/DI and/or NASA will be available for a continuation of efforts in the area.

4.3 ILLIAC IV LANGUAGE DEVELOPMENT FOR THE PHASE II SYSTEM

4.3.1 Introduction

A six month study, described in preliminary terms in the last progress report, culminated with a report on the IDOL (ILLIAC Data Oriented Language). That language is designed to make flow of control and working set information readily available to the compiler. The information would be used to produce programs which interact smoothly with the memory hierarchy to provide a steady flow of data to and from ILLIAC IV with a minimum of "page faulting."

4.3.2 IDOL

IDOL (ILLIAC Data Oriented Language) is designed to facilitate program construction for the entire ILLIAC system. It has two portions--a data language in which to express data movement between the phase II memory hierarchy and ILLIAC, and an algorithmic language to be used for ILLIAC processing. In this report, we shall emphasize the data language, for although languages for ILLIAC have often been studied, the problem of data management for a memory hierarchy has received relatively little attention.

In handling data management for ILLIAC IV, one quickly realizes that it is really a virtual machine, since the data required for an ILLIAC problem typically exceeds the capacity of the core memory (PEM). As in other virtual memory systems, the fundamental problem is efficient use of the memory hierarchy so that CPU usage is maximized while minimizing channel traffic. The problem is aggravated on ILLIAC IV for several reasons.

First of all, on other virtual systems multiprogramming is possible, so that one program may run while the others wait. On ILLIAC IV, multiprogramming is impractical because of the extremely small core memory in comparison to the processing power. Also, ILLIAC IV is so fast compared to the rotation

time of its disk that even fast transmissions from the disk, if not timed exactly right, can cause serious loss of efficiency, while the time lost by waiting for data from lower levels of the hierarchy is particularly disastrous. Finally, unlike other useful virtual systems, ILLIAC IV has no special paging or even memory protection hardware, which imposes a greater software overhead than in other systems if page faulting is to be relied upon.

In the past, attempts to have the proper data present when needed have centered on various ways to determine the current working set for a program in terms of what had been accessed in the recent past. The efficiency of such schemes varies with different programs, but is always well below optimum.

The primary focus in the design of IDOL is that working set determination and sequencing are a function of the compiler; i.e., IDOL is designed to make the appropriate working set for each segment of an algorithm, and also the flow of control between these segments, apparent to the compiler. As a consequence, the compiler can generate a set of programs which will make maximal use of the available ILLIAC resources. The compiled programs will execute on the ILLIAC IV--PDP-10 multiprocessor with appropriate PDP-10 companion processes formulated in response to one unified source program. Moreover, the programs produced will run under the standard ILLIAC operating system and will, therefore, serve to augment the basic system.

4.3.3 Types of Problems

IDOL is designed for efficient movement of data through the phase II hierarchy, even under changing allocations of space on the various devices, and is primarily for problems in which ILLIAC execution proceeds in an orderly manner through a large data base, processing (and possibly updating) a well

defined part of the data base in each step, with many passes possible through the data base. In reading papers on problems proposed for ILLIAC IV and in discussions with probable ILLIAC users, we have determined that, for the problems we have seen, the majority of the input/output fits the pattern for which IDOL was designed. For the few cases in which the sequence of (data dependent) I/O requests cannot be determined in advance, explicit I/O statements will be allowed in the ILLIAC language portion of IDOL.

4.4 ECONOMIC RESEARCH GROUP

4.4.1 STEP I

During this period, the Economic Research Group estimated all the component models of STEP I and worked toward completing the interconnection of these models. Initial planning and design for enlarging STEP I from 218 economic activities and 86 industries to 400 activities and 376 industries has been completed and a proposal for support of this work has been submitted to NSF RANN. Work under ARPA support on STEP I should be completed by the end of December, 1972.

4.4.2 STEP II

Planning and design for STEP II has begun. Support for this work will be sought from NSF RANN and the Department of Transportation and will involve no cost to ARPA.

4.4.3 MEASURE

MEASURE, a user-oriented operating system involving network computers (PDP-10, B6700, and 360/91), has been substantially expanded to provide a wide range of library maintenance, editing, mathematical, and statistical routines. The first experiments in using the PDP-10 to establish a communications link to the B6700 have been successful and code to establish the link to the 360/91 has been written. An efficient matrix computations package for the 360/91 has been written and a statistical package is being written. The first experiments in computer initiated transfer to computations from the B6700 to the 360/91 will be carried out in November. Initial benchmark calculations suggest that interactive jobs on the B6700 cost about one-fifth as much on the B6700 as in the 360/91 and large matrix computations on the 360/91 will cost between one-eighth and one-fourth as much as the B6700. The use of MEASURE with the 360/91

being used invisibly for large computations can be expected to realize substantial cost savings over either the B6700 or the 360/91 used alone. When fully implemented, MEASURE will permit naive users to use STEP I for forecasting over the network and MONICA for general numerical and statistical computations at minimum computational cost.

4.5 NETWORK SYSTEMS GROUP

4.5.1 Introduction

The Network Systems Group has research responsibility in the area of development of local basic computer systems and services to serve the need of the Center staff and user community, and for the development of hardware and software systems for Network utilization and implementation. There are four project areas to be reported on at this time:

- a. Project to Interface the Burroughs B6700 to the ARPA Network
- b. ARPA Network Terminal System (ANTS)
- c. Center Graphics Support
- d. Network Graphics Effort

4.5.2 Project to Interface the B6700 to the ARPA Network

During the period, a re-evaluation of the software effort needed to bring the UCSD (University of California at San Diego) Burroughs B6700 system onto the Network indicated that in order to provide Network access in the shortest amount of time, the NCP effort perviously accomplished by the Center would be used. Additional effort was therefore applied and the NCP project re-instituted.

The first version of the NCP was completed, debugged, and installed in the UCSD system utilizing a temporary connection to the Network, via a 3000-baud line direct to the UCLA Network IMP. This enabled the San Diego system to come on the Network around August 1, 1972.

The Center staff member associated with this area of research left the Center and is now employed by the UCSD Computer Center. Therefore, all further work on interfacing Burroughs B6700's to the Network will be accomplished at UCSD with no further support from the Center.

4.5.3 ARPA Network Activities

4.5.3.1 ARPA Network Terminal System (ANTS) Development

During the reporting period, development of ANTS continued to be affected by the performance of the in-house Burroughs B6700 system. With the advent of the B6700 system at UCSD becoming a full Network site, the in-house B6700 was removed from service on June 30, 1972.

Further development efforts on ANTS were directed at stabilizing the current system, now known as MARK I, with capability at the minimum TELNET level and including remote job entry service to the UCLA 360/91.

In addition, several "Kludge" software patches were added which allow the transmission, for experimental purposes, of graphics images back from UCSD, Rand Corporation's TENEX system, and 360/91, to both the Gould graphics plotter and the Computek storage scopes.

Remaining efforts of the group during this period were to complete software necessary to operate with the UCSD Computer Center via the Network and provide the development group with PEESPOL support for developing MARK II ANTS in the next reporting period.

The MARK II system will implement all available Network protocols such as low level graphics protocol, file transfer protocol, full TELNET protocol, and remote job entry service protocol. In addition, all of the peripheral devices such as DECTapes, mag tapes, Gould printers, card readers, disks and graphics displays will be fully incorporated and the system brought to a completed status.

4.5.3.2 ARPA Network Usage

During the reporting period, usage of the Network continued at an increasing rate. TSO was added at the 360/91 and access to it was gained by Center members.

As of August 1, 1972, the UCSD B6700 system was available as a Network Host and the Center programming efforts on the machine were transferred to that site. The B6700 at San Diego was brought to full status of operations with such Center provided systems as MONICA, WARIS, the PEEBOL compiler and ANTS development software packages, and the use of the ILLIAC IV software packages maintained there by Ames.

4.5.3.3 Further Installations of ANTS Systems on the Network

During the reporting period, further contacts were made with additional sites desiring information as to the availability of an ANTS system to provide those sites with access to the Network. An investigation is under way regarding the feasibility of establishing a number of sites on the Network for the Army Materiel Command and providing the various sites with access to Network for such service sites as UCLA 360/91, ILLIAC IV, and several AMC service sites such as the CDC-6600 installation at Ft. Belvoir, Virginia.

In addition, the system utilization measurement group at Lawrence Radiation Laboratories, Livermore, California, has elected to procure an ANTS system to facilitate their gaining access to the Network and to non-Network service sites in order to study the utility of access to those systems and to evaluate the services those systems supply.

Negotiations have been under way with Digital Equipment Corporation for the procurement of additional hardware at the Center, based on the advanced concepts ILL-11 model 45 processor and memory system, to provide Center research personnel with an advanced design access system to the ARPA Network.

During the next reporting period, a basic design of the system will be undertaken and an implementation schedule for its production will be developed. This enlarged advanced system will provide the Center with a mechanism for studying Network operations in the area of process-to-process communications, software resource sharing, processor service sharing, etc.

4.5.4 Graphics Support for Center Projects on the Burroughs B6700

With the advent of the UCSD B6700 system as a service site on the Network, effort originally terminated on the Center's B6700 system for providing graphics support software packages to Center personnel was reinstituted. Initial efforts were directed at bringing previously developed in-house systems up to full operational status to be used remotely between San Diego and the local ANTS system. The extent and conclusion of this effort will be documented in the next reporting period.

4.5.5 Network Graphics Efforts

4.5.5.1 Network Graphics Protocol

During the reporting period, the first level graphics protocol was agreed to by members of the Network Working Group. No further meetings have transpired during this time to enlarge upon or elucidate experiences in the production of this protocol.

Efforts in the Center have been directed at developing the capability of sending Network graphics protocol from UCSD, Rand TENEX, and the Model 360/91 at UCLA. Initial experiments in the use of this protocol were moderately successful, due mainly to the problems in returning images directly to ANTS supported graphics peripherals.

4.5.5.2 Laboratory for Atmospheric Research Support

Support of the University's Laboratory for Atmospheric Research, under the direction of Professor Ogura, was limited during the period to the study, specification, and acceptance of bids by graphical plotter vendors for the purchase of a drum plotting device to be attached to the Center's ANTS system. This proposed device would provide high-quality incremented graphics plotting capability for both Laboratory and Center personnel. During the next reporting period, it is proposed that this device be procured, installed, and software for the generation of graphical images be developed at UCSD, Rand and on the UCLA model 360/91.

4.6 ADMINISTRATION

4.6.1 Introduction

Due to a last minute budget reduction of \$100,000 by ARPA, significant changes were made at the Center, resulting in a decrease of personnel.

4.6.2 Fiscal Status

Actual expenditures through 31 March 1972: \$723,965

Estimated expenditures and obligations for the 6-month period covered by this report (1 April - 30 September 1972):

April	139,045	
May	121,430	
June	165,399	
July	109,965	
August	153,025	
September (estimated)	<u>94,200</u>	
	783,064	<u>783,064</u>

Total estimated expenditures and obligations through
30 September 1972:

\$1,507,029

REFERENCES

- [1] Wilkinson, J. and Reinsch, C., Handbook for Automatic Computation, Vol. 2, Linear Algebra, Springer-Verlag, 1971.
- [2] Wilkinson, J., The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1965.
- [3] Paige, C. C., "Practical Use of the Symmetric Lanczos Process with Reorthogonalization", BIT, 10, pp. 183-195, 1970.
- [4] Golub, G. H., Underwood, R., and Wilkinson, J. H., "The Lanczos Algorithm for the Symmetric $Ax = \lambda Bx$ Problem", Computer Science Department, Stanford University, STAN-CS-72-270, March 1972.
- [5] Cohen, S., Vincent, C. M., "An Introduction to SPEAKEASY", Argonne Physics Division, Informal Report PHY-1968E, Dec. 1968, (Revised June 1972).
- [6] Gill, P. E. and Murray, W., "A Numerically Stable Form of the Simplex Algorithm", Tech. Report No. Maths. 87, National Physical Laboratory, Teddington, 1970.
- [7] Saunders, M. A., "Large-Scale Linear Programming Using the Cholesky Factorization", Computer Science Department, Stanford University, STA-CS-72-252, January 1972.
- [8] Warshall, S., "A Theorem on Boolean Matrices", JACM, 9, pp. 11-12, 1962.
- [9] Mulligan, G. D., "Algorithms for Finding Cliques of a Graph", Tech. Report No. 41, Department of Computer Science, University of Toronto, 1972.
- [10] Belford, G., "Uniform Approximation of Vector-Values Functions with a Constraint", Math. Comp., 26, pp. 487-492, 1972.
- [11] Belford, G., "Vector-Valued Approximation and its Application to Fitting Exponential Decay Curves", Math. Comp., To appear.
- [12] Rice, J., "Algorithms for Chebyshev Approximation by $ab^x + c$ ", J. SIAM, 9, pp. 571-583, 1961.

DOCUMENTS AND PUBLICATIONS

- R. H. Bezdek, "Economic Research Group Working Paper No. 6: Simulating the Employment Impacts of the Urban Coalition's Counterbudget," CAC Document No. 16, University of Illinois at Urbana-Champaign, October 15, 1971.
- H. J. Bouknight, "ANTS - A New Approach to Accessing the ARPA Network," CAC Document No. 47, University of Illinois at Urbana-Champaign, July 1, 1972.

- W. J. Bouknight, "The ILLIAC IV System," Proceedings of the IEEE, Vol. 60, No. 4, pp. 369-388, April 1972.
- S. A. Denenberg, "ANTS - A New Approach to Accessing the ARPA Network," CAC Document No. 47, University of Illinois at Urbana-Champaign, July 1, 1972.
- _____, "A 10-Page Description of the ILLIAC IV System," CAC Document No. 30, University of Illinois at Urbana-Champaign, October 12, 1971.
- _____, "Getting Started on the ARPA Network Terminal System (ANTS)," CAC Document No. 42, University of Illinois at Urbana-Champaign, August 1, 1972.
- _____, "The ILLIAC IV System," Proceedings of the IEEE, Vol. 60, No. 4, pp. 369-388, April 1972.
- D. M. Grothe, "PEESPOL Overview," CAC Document No. 31, University of Illinois at Urbana-Champaign, July, 1971.
- D. J. Hopkin, "Information Retrieval and the ILLIAC IV," CAC Document No. 44, University of Illinois at Urbana-Champaign, May, 1972.
- R. J. Lermitt, "A Linear Programming Implementation," CAC Document No. 46, University of Illinois at Urbana-Champaign, October 1, 1972.
- L. M. McDaniel, "Symmetric Decomposition of Positive Definite Band Matrices and the Corresponding Solution of Systems of Linear Equations on ILLIAC IV," CAC Document No. 34, University of Illinois at Urbana-Champaign, July 1, 1972.
- D. E. McIntyre, "The ILLIAC IV System," Proceedings of the IEEE, Vol. 60, No. 4, pp. 369-388, April 1972.
- K. Miura, "The Block-Iterative Method for ILLIAC IV," CAC Document No. 41, University of Illinois at Urbana-Champaign, May 10, 1972.
- S. A. Pace, "An ILLIAC IV Gaussian Elimination," CAC Document No. 40, University of Illinois at Urbana-Champaign, September 1, 1972.
- J. M. Randal, "The ILLIAC IV System," Proceedings of the IEEE, Vol. 60, No. 4, pp. 369-388, April 1972.
- R. M. Ray, "BLOCKS: A FORTRAN IV Program for Plotting Planar Projections of Three-Dimensional Block Models," CAC Document No. 43, University of Illinois at Urbana-Champaign, June 15, 1972.
- A. H. Sameh, "The ILLIAC IV System," Proceedings of the IEEE, Vol. 60, No. 4, pp. 369-388, April 1972.
- D. L. Plotnick, "The ILLIAC IV System," Proceedings of the IEEE, Vol. 60, No. 4, pp. 369-388, April 1972.

THESES

- J. H. Ericksen, "A Survey of Iterative Methods for Solving Poisson's Equation and Their Adaptability to ILLIAC IV," Ph.D. Thesis directed by D. L. Slotnick, June, 1972.
- R. J. Lermitt, "Numerical Methods for the Identification of Differential Equations," Ph.D. Thesis directed by D. L. Slotnick, June, 1972.
- R. B. Wilhelmson, "The Numerical Simulation of a Thunderstorm Cell in Two-and Three-Dimensions," Ph.D. Thesis directed by D. L. Slotnick, June, 1972.

5. THEORY OF DIGITAL COMPUTER ARITHMETIC

(Supported in part by the National Science Foundation under Grant Number US NSF GJ 813.)

5.1 Study of Logical Organization for LSI Implementable Arithmetic Units

In this quarter, various methods of distributing control in the digit processing modules (DPM) to find an optimal balance between the global control and local control in the DPM's were studied. A mixture of associative techniques in the global control and microprogramming techniques in the local control look promising. However, no definitive conclusions about the extent of each have been reached and work is being continued towards this end. At this stage, it almost seems certain that we shall need some amount of central global control and that the control cannot be totally decentralized in the DPM's, especially for the division process.

Further studies should give us a better insight into decentralizing the control of arithmetic algorithms in the DPM's.

(L. N. Goyal)

5.2 Automatic Evaluation of Some Elementary Functions Using Microprogramming

During this quarter, the work on 'Automatic Evaluation of Some Elementary Functions Using Microprogramming' was continued. The problem being studied is whether our previous results (e.g., the speed) can be improved by slightly modifying the Burroughs D-machine (e.g., by adding one or two more registers). The other problem being studied now is how much total micro-memory is needed for

implementation of all the algorithms for division, multiplication, logarithm, exponential, square root, and other trigonometric functions. For example, can the micro-memory needed be reduced by sharing part of the programs among the algorithms for evaluating different functions? Some progress has been achieved in these directions.

(Peter D. Ting)

5.3 Implementation of Continued Product Algorithms

In this quarter an investigation of a possible refinement of continued product algorithms and corresponding implementations was continued. Also, the applicability of a mixed normalization technique was studied further. The final report on radix 16 automatic evaluation of some functions, in the form of master's thesis, was completed. Also, a report, describing analogous algorithms and implementation for the radix 10 approach was written.

(M. D. Ercegovic)

6. SWITCHING THEORY AND LOGICAL DESIGN

(Supported in part by the National Science Foundation under Grant Number U.S. NSF-GJ-503A #1.)

J.J Mora-Touar finished a Master degree thesis on the effect of additional inequalities on the computation time of logical design by integer programming. He concluded that some types of additional inequalities were very effective but other types tend to have adverse effects depending on the gate types.

Transformation techniques to improve given networks were extensively tested by computer during this period by J.N. Culliney, Y. Kambayashi and H.C. Lai.

The manual of logical design of optimal networks by implicit enumeration method based on the all-interconnection formulation was finished.

M. Jamieson started to look at basic reading material for his Master degree thesis.

S. MUROGA

During this quarter, we continued experiments with the different versions of our Phase II program. Also, starting with different initial networks (realizing the same output function) obtained by different methods, Phase II was applied to transform the redundant initial networks into more nearly-optimal networks. Interestingly, the results did vary significantly (for some functions) depending on which initial network was used as a starting point for the Phase II procedure.

Work also began on writing DCL Reports to document the procedures and computer programs (which realize the procedures) developed so far.

During the last quarter period we developed algorithms for NO* (HAND) network transformations which uses an idea of error compensation. These algorithms were programmed and tested by computer. By these experiments many improvements and modifications were made to the procedure.

The effectiveness of the procedure is being examined in a logical design system consisting of the following two steps.

- (1) Synthesis of a logic network which realizes a given function ((a) The first solution of branch-and-bound method; (b) A universal network; (c) A three-level network obtained by Gimpel's procedure; (d) A simple procedure to synthesize a three-level network; (e) Others).
- (2) Application of the network transformation procedure using error compensation ((a) Single path method; (b) All path method).

In step (1), a different synthesis procedure produces a different network in a different calculation time (we have computer programs for (a), (b), (c), and (d)). In step (2), (b) is more powerful than (a) because it examines all possible transformations. But the calculation time required for (b) is much longer. Experiments were done for various combinations of procedures in steps (1) and (2) because these results give useful information for constructing a logic design system based on network transformations. Some of the results obtained are shown below.

(a) Single path method.

The following five-variable functions were selected using a random number table and their optimum networks are not yet known.

The results show that the final networks and calculation times depend on the nature of the initial network.

Functions	No.1	No.2	No.3	No.4			
Initial network is the first solution of the B&B	Initial network	19 73	20 69	29 99	17 58	gates connections	
	Final network	12 39	15 48	12 33	13 39	gates connections	
	Time	4.43	9.33	12.15	14.90	seconds	
	Initial network	26 100	25 100	25 105	22 92	gates connections	
		Final network	14 45	13 42	10 35	11 36	gates connections
		Time	15.40	9.29	5.41	7.01	seconds

(b) All path method.

When we applied the procedure to the network obtained by the first solution of the branch-and-bound method for function No. 4 of the above table, a network with 10 gates and 35 connections was obtained after 599.02 seconds of calculation time (by single path method a network with 13 gates and 39 connections was obtained after 14.9 seconds of calculation time).

For the above examples optimum networks are unknown. In order to measure the actual ability of the transformation procedure we must design networks for functions whose optimum networks are already known. Many kinds of experiments are now being undertaken.

For better transformations we introduced an "order of error". If we compensate an error of smaller order then the probability of compensation of the errors of the network outputs is higher. So a transformation using this concept will be able to compensate errors more efficiently, resulting in better transformations. A primitive version of the program based on this procedure was prepared by H.C. Lai (see Quarterly Report).

We began to document our network transformations and network synthesis procedures using wired-logic gates.

(Y. Kambayashi)

The NOR network transformation program by error compensation procedures was debugged and tested. During debugging and testing some possible improvements were found, and therefore several versions of this program were developed and examined. Although the tested examples are not sufficient for thoroughly evaluating these different versions, some of the new versions are definitely better than the old one. Another improvement of this procedure, which requires a much greater effort to be implemented into the program was proposed (See Quarterly Progress Report, July-Aug.-Sept., 1972 by Y. Kambayashi). The primitive version of this newly proposed procedure was coded and debugged. The initial results are not so good as expected. This does not mean, however, that the new version is less powerful but that more effort should be put on debugging and improving the primitive version.

(H.C. Lai)

The user's manual for the all-interconnection formulation network synthesis program has been completed. Several errors in this program were found and corrected. These errors did not affect any of the results obtained previously but could have affected future problems run on the program. Gimpel's program for finding optimal 3-level TANT networks was used to solve some sample functions. These networks will be used as initial solutions in the transformation programs of H.C. Lai, Y. Kambayashi and J. Culliney.

(K. Hohulin)

(Supported in part by the National Science Foundation under Grant Number
US NSF GJ 27446)

The following is a collection of related work aimed at improved designs for computer and software systems. We are interested in parallel and pipeline processors, small primary memories, effective use of rotating memories, and some questions concerning user languages for problems including typical FORTRAN type calculations, simulation languages, and a variety of file processing problems.

(D. Kuck)

7.1 FORTRAN Parallelism Detection - (S. C. Chen, R. Strebendt, and R. Towle)

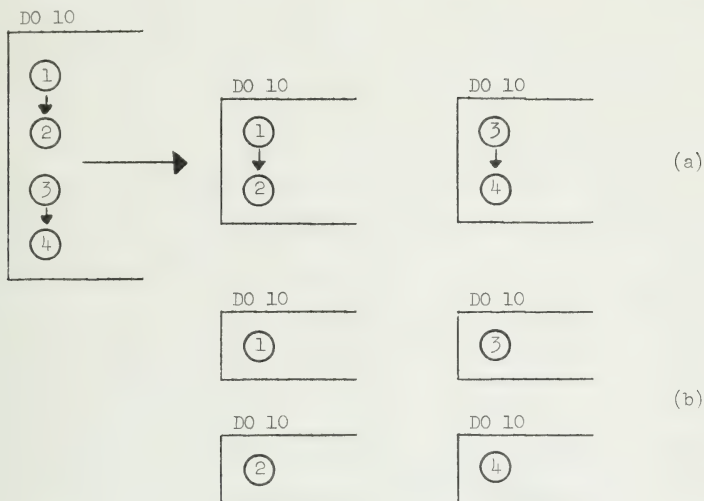
The original DO subprogram, incorporated with all other subprograms in the analyzer, has been tested for many real FORTRAN programs and the results of that measurement have been collected.

During this period, a new strategy of cutting/splitting a DO loop in order to take advantage of forward substitution across iterations has been implemented. The algorithm is outlined as follows:

1. Find the dependence graph of the loop.
2. Separate this graph into complete disconnected subgraphs and execute them in parallel.
3. If, within each subgraph, all statements are assignment statements, then we perform forward substitution across iterations to exploit parallelism by using tree-height reduction techniques (in ASSIGN subprogram). If there are inner DO loops, then forward substitution and tree-height reduction are performed only to those blocks of assignment statements inside each inner loop. And then, using the same strategy, we perform steps (1), (2), and (3), until we exploit all parallelism inside each inner loop. After that, we can concatenate all inside assignment blocks and DO blocks to perform sequentially with respect to the outside DO loop.

The following simple example will show the difference between the old

(b) and new (a) result:



The algorithm has been implemented and tested for many real FORTRAN programs. The result shows that both horizontal and vertical schemes give a good speed-up. The horizontal scheme tends to facilitate the extraction of type-1 parallelism while the vertical scheme helps to find type-0 parallelism. There is no clear way to tell which scheme is better, since it depends on the characteristic of the DO loop.

Here is a rough statistic from a sample of about 28 DO loops contained in 8 randomly selected programs: 17 cases yield the same result for both schemes; 9 cases have higher speed-up for vertical scheme than horizontal scheme, but there are 4 cases among them which obtain the higher speed-up at the expense of lower efficiency; the remaining 2 cases give a lower speed-up but higher efficiency for the vertical scheme. In examining the codes of these programs,

we can roughly say that: if there is no internal loop inside a completely disconnected subgraph of β -graph, we can use the vertical scheme, otherwise we use the horizontal scheme for that subloop. These combining schemes probably will give a better result if we are only concerned about speed-up.

Several relatively minor changes were made in the assignment statement analyzer. Among these changes were:

1. We altered the handling of assignment statements having a scalar variable (rather than a subscripted variable) on the left hand side occurring in DO loops. The algorithm now takes into account any dependency, direct or indirect, on the variable of iteration.

2. The functions, used to extrapolate the results for a block which has an iteration limit greater than that which the analyzer is capable of handling, were refined. The analyzer now can recognize a linear or non-linear recurrence relation and apply the appropriate extrapolation function.

3. Data structures within the analyzer were changed to allow the processing of some FORTRAN codes which previously exceeded the capacity of the analyzer.

4. Many minor bugs were fixed and traps were added to catch particular cases in which the analyzer previously ran off of the end of a table without giving any indication of this until the program ABENDED some time later.

5. The size of the program and its speed were improved by recoding portions of it to make better use of the PL/I language than it did originally.

Approximately 30 new FORTRAN programs were analyzed. Some routines have been modified to allow comparison of BAS analysis methods. Future plans include analyzing more FORTRAN programs and a detailed comparison of methods of analyzing BAS.

7.2 Theoretical Bounds on Parallelism - (K. Maruyama)

During the months of August and September, parallel algorithms to evaluate arithmetic expressions have been studied and a paper entitled "The Parallel Evaluation of Arithmetic Expressions," has been written. Following is the abstract of the paper.

In this paper we consider the parallel evaluation of arithmetic expressions of n distinct variables with addition, subtraction, multiplication, and division operations. We first show that any such expression can be evaluated in at most $4\log_2 n + O(1)$ steps if $c_1 n$, $c_1 \leq 1.5$, processors are available. Second, we show that any continued-parenthesis form of n distinct variables can be evaluated in at most $2\log_2 n + O(1)$ steps if $c_2 n$, $c_2 < 1$, processors are available. Third, we show that any polynomial form of n distinct variables can be evaluated in at most $\log_2 n + \sqrt{2\log_2 n} + O(1)$ steps if $c_3 n$, $c_3 < .5$, processors are available.

We also deduce some results, on each of the above cases, which apply when a fixed number of processors are available.

Information Retrieval and File Processing - (L. Hollaar, W. Stellhorn, and J. Rinewalt)

Due to a large number of component failures, relatively little progress has been made this quarter with the physical installation of the D-Machine. At the present time parts orders are outstanding for the card reader, the line printer, and one of the mainframe power supplies. In addition, tests are pending for repair work on part of the user memory. At the time of the power supply failure, which brought all further systems work to a halt, the bootstrap and loader programs had both been tested extensively, and a major portion of the microcode which supports the text-searching facilities (S-Language) had been

tested. In addition, printer interface hardware and software have been completed (but not tested) as have a dump routine on S-Language programs and certain extensions to the original retrieval system. It is anticipated that when the power supply is restored extensive testing can be performed on the interactive retrieval program and the input-output interfaces.

During this quarter, more attention than before has been directed toward the design of specialized computers for information retrieval. Ideas under discussion include practical methods for scanning the full text of large data bases in reasonable amounts of time and for using special purpose hardware to perform term coordination in order to eliminate a serious bottleneck in existing large retrieval systems.

7.3 Switching Networks - (D. Lawrie)

During this quarter investigation of various switching networks was begun. Primary interest is centered on networks whose properties make them useful for interconnecting various components in a computer system (e.g., memories, processors). Preliminary results indicate that modifications of networks proposed by Benes, Clos, Batcher, and others, might be suitable. A paper on this subject was presented at Compcon 72, IEEE Computer Society Conference, in San Francisco, September, 1972.

Papers Published in this Quarter

Edward W. Davis, Jr., "Concurrent Processing of Conditional Jump Trees," Compcon 72, IEEE Computer Society Conference, San Francisco, Sept. 1972.

David E. Gold, "Applications of Some Switching Network Results to Dynamic Allocation of Memories in a Hierarchy," Compcon 72, IEEE Computer Society Conference, San Francisco, Sept. 1972.

D. J. Kuck, D. H. Lawrie, and Y. Muraoka, "Interconnection Networks for Processors and Memories in Large Systems," Compcon 72, IEEE Computer Society Conference, San Francisco, Sept. 1972.

D. J. Kuck, and Y. Muraoka, "Fast Computers from Slow Parts," Compcon 72, IEEE Computer Society Conference, San Francisco, Sept. 1972.

8. COMPUTER SYSTEMS ANALYSIS

(Supported in part by the National Science Foundation under Grant No. NSF GJ 28289)

The goal of this research is the development of analytical tools for system modeling and analysis of real time computer networks. Priority assignment and job dispatching rules for a geographically distributed computer network are being investigated.

8.1 Computer Network Modeling (L. Mills and W. McKinney)

Our efforts have been concentrated on developing a viable analytical model for a computer network. Our view of the network is much the same as an outside observer's. We see jobs coming in to each center i at a rate α_i and eventually leaving, either through process completion at rate μ_i or through transmission to another center (for whatever reason) at rate Ω_i . Internal to this structure is the priority assignment, if any, and the load leveler for that particular center. Each center is considered to be of equal, although not necessary homogeneous, capacity. In our initial analysis, Poisson arrival rates and exponential service rates have been assumed. We have developed separate queueing theory models for the two server and three server systems for the cases in which 1) the servers are assigned jobs randomly and 2) the servers are assigned jobs according to a specified heirarchy.

8.2 Center Throughput Analysis (S. Mamrak and F. Salz)

A GPSS simulator for the I.B.M. 360/75 operating system configuration at the University of Illinois was designed and written. In order to verify the accuracy of the simulation model, statistics were gathered from

the operating system, and frequency distributions for various arrival and service rates were constructed. These distributions then became the input functions for the GPSS model, and the simulation was run with what was assumed to be a representative job stream. The statistics gathered include CPU usage, number of I/O requests, time in O.S., and core region requests, tabulated by job, job class, and job step. Given the distributions for CPU time, I/O requests, and core request, the simulation was run, and statistics on simulated time in O.S. were compiled. The time in O.S. for both the actual system and the simulation were graphed, and the results confirmed the valid operation of the simulator.

A priority scheme has been devised with a view to minimizing the mean flow time of a set of jobs on the 360/75, while maintaining a given level of CPU and memory utilization. The methodology for developing the priority scheme is one of using a GPSS simulation of the performance of the 360/75 to both modify and measure system parameters before and after the proposed changes. To date three representative job streams for the 360 have been reproduced for the GPSS simulation. The simulation is presently being used in conjunction with the first of these job streams to measure the relationship between a job's core requests and its respective waits for core. This relationship is not directly measured by the actual 360/75, but is very readily available from the simulation. The hypothesis that a job's total wait time for core is approximately equal to its wait time for its largest core request is also being tested. The information from these measurements will be used to assign a job an initial static priority as it enters HASP. Further changes and measurements will then be made to assign subsequent dynamic priorities to the job.

Publications

E. K. Bowdon, Sr., "Networks Computer Modeling," Proceeding of the ACM, August, 1972, pp. 254-264.

F. R. Salz, "A GPSS Simulation of the 360/75 under HASP and O.S. 360," Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, Report No. UIUCDCS-R-72-528, June, 1972.

W. J. Barr, Master of Science Thesis, "Cost Effective Analysis of Network Computers," Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, Report No. UIUCDCS-R-72-538, August, 1972.

J. T. Fitzgerald, Master of Science Thesis, "Load Regulation and Dispatching in a Network of Computers," Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, Report No. UIUCDCS-R-72-537, August, 1972.

(E. K. Bowdon)

9. NUMERICAL ANALYSIS

The finite element method (FEM) of approximation of the solution of boundary value problems for elliptic equations and boundary-initial value problems for parabolic equations reduces the problem to a linear system of algebraic equations,

$$(1) \quad Au = s.$$

Essentially, the only technique that engineers grudgingly use to solve (1) is a direct method, that is, some form of Gaussian elimination. Iterative methods are generally shunned, because of slow convergence. Direct methods seem preferable because they are simple to code and the code executes. For these reasons, engineers and scientists employ direct methods, but complain of code that absorbs the entire memory for lengthy runs. A crisis could develop from these difficulties. The use of currently popular direct methods limits the range of solvable problems. The satisfactory solution of three dimensional problems for, say, one thousand time steps must await the development of new techniques.

Several approaches are being studied. According to Martin Schultz, the use of high order basis functions yields an accurate approximation from a low order linear system. Schultz's conclusions are limited to problems defined over a square domain for which the solution is reasonably smooth. Nevertheless, his work demonstrates the practical worth of high order basis functions. Unfortunately, three dimensional problems, problems with singularities, and problems with difficult, irregular domains still seem to yield large linear systems.

George Fix and K. Larsen have extended the mathematical analysis of successive-over-relaxation (SOR) to finite elements system and showed that if n^2 is the number of unknowns in (1), the solution of (1) may be determined to a specified level of accuracy with $O(n^3)$ arithmetic operations. This is considerable economy over conventional direct methods. Their work is subject to restrictions on the domain of the problem, but likely is correct for more general problems.

Despite the improvement Fix and Larsen have demonstrated, it is not clear that SOR is superior to all forms of Gaussian elimination. Their conclusions show that for the problems they study, SOR is superior only to conventional applications of Gaussian elimination. By ingeniously reordering the nodes of the finite elements, Alan George has shown how to solve (1) with $O(n^3)$ operations. Thus, direct methods remain as fast as SOR after George's analysis. A full understanding of the comparative merits of SOR to George's technique awaits further study. Iterative methods still require less storage, an important advantage for now but perhaps not in the future. George's analysis is, as the work of Fix and Larsen, subject to restrictions on the domain.

This completes a brief account of the need for efficient methods to solve finite element linear systems and the work elsewhere to meet this need. I will conclude by mentioning our work on the solution of finite element systems. Y. J. Kim has coded a factorization procedure for a mildly complicated triangularization of a rectangular domain. The matrix of the resulting system is irregular and large, but sparse. This system can be solved in 30 steps with relative error less than 10^{-10} using the adaptive-Chebyshev-factorization algorithm of Martin Diamond. No direct comparison has been made to SOR. This experiment has been compared to

Richardson's method for which the relative error is reduced to less than 1. Because the rate of convergence for SOR is somewhat more than twice the rate of convergence for Richardson's method, for finite difference systems, we infer that SOR might reduce the relative error to less than 10^{-2} . This experiment yields the same result observed for the solution of a very regular finite element system obtained from a triangularization using congruent equilateral triangles. There is no apparent restriction on the method due to irregularity of the matrix, although studies are incomplete. Further work is required for implementing the method to general problems, extending it to mixed and Neumann boundary conditions, and developing the mathematical properties of factorizations.

(Paul Saylor)

10. COMPUTER LANGUAGES FOR MATHEMATICS AND NUMERICAL ANALYSIS

The OL/2 language is an algorithmic array language with algebraic and geometric properties. The algebraic properties are derived from the field of linear algebra while the geometric properties are based upon compact data types that are largely independent of the algebra. The combination of the algebraic and geometric properties together with other array operations allows users to construct efficient array algorithms. The OL/2 language and its compiler provide a means for studying array languages, array compilers, array algorithms, and parallel computation.

The current compiler (version 3) is now available to the university community. An OL/2 user's guide and a DCL report giving an overview of the language will be available during the last quarter of 1972.

10.1 Array Expressions in OL/2

The compiler modules which are concerned with the compilation of array expressions is described in the thesis of Dale Jurich, entitled "An Approach to the Compilation of Array Expressions in the OL/2 Language." An abstract and reference to the DCL report (UIUCDCS-R-72-550) is provided in the appropriate section of this issue of the Quarterly Progress Report.

This report describes the parsing of array expressions, the abstract and concrete representations of the appropriate tree structures. The syntax and semantics of OL/2 array expression module are documented in the appendix of the report.

10.2 Compiling and Executing OL/2 Programs

Version 3 of the OL/2 compiler is implemented on the IBM 360. The compiler is a one pass compiler and therefore requires a fairly large region of 290k bytes. The JCL for compiling and executing an OL/2 program is given below:

```
/*ID PS= ...
/*ID TIME=(0,30),IOREQ=2500,REGION=290K
//      EXEC OL2
//OL2.SYSIN DD *

      ( OL/2 PROGRAM )

//GO.SYSIN DD *

      ( DATA )

/*
```

The OL/2 program may be keypunched using the standard 60 character set on the 029 keypunch, or it may be constructed at a PLORTS terminal. If an IBM 2741 terminal is available, as in Room 94 DCL, you may use an 88 character set which allows a slightly nicer set of symbols for some operations; it also allows lower case letters and upper case letters as distinct symbols. If the 88 character set is to be used, then the above JCL must be altered and communication through PLORTS is also altered. The OL/2 group in Room 94 DCL will provide the necessary information.

10.3 Examples of OL/2

In order to indicate a few of the characteristics of the OL/2 language, we present several examples. These examples

were constructed at the terminal using the 88 character set,
and the programs were executed.

EXAMPLES: MAIN PROCEDURE;

N = 4;

LET A AND B BE MATRICES OF ORDER (N); LET x, y,
z, AND r BE VECTORS OF ORDER (N); LET p AND q BE SCALARS;
INPUT x, y, z, A, B; OUTPUT x, y, z, A, B;

p = 3.14;

q = 2.71;

SIMPLE_ARRAY_EXPRESSIONS:

r = z - A*x;

OUTPUT r';

y = y - (r,r)/(A*r,r)*r;

OUTPUT y';

z = ((A'*x + y'*B*x*x*y)'*A)';

OUTPUT z';

y = 3*SIN(||A|| - ||B||)*A*x + 2*(x + y);

OUTPUT y';

IF || A - B || > 0.013 THEN q = 1.2; ELSE p = 2.3;

OUTPUT q, p;

END EXAMPLES;

!X !	!Y !	!Z !
-1-	-1-	-1-
1	5	9
2	6	10
3	7	11
4	8	12

!A *	*	*	!	!B *	*	*	!
-1-	-2-	-3-	-4-	-1-	-2-	-3-	-4-
13	14	15	16	29	30	31	32
17	18	19	20	33	34	35	36
21	22	23	24	37	38	39	40
25	26	27	28	41	42	43	44

	!K_	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	!
		-1-		-2-		-3-		-4-		
-1-		-3.730000E+02		-4.760000E+02		-5.790000E+02		-6.820000E+02		
	!Y_	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	!
		-1-		-2-		-3-		-4-		
-1-		9.497810E+00		1.173983E+01		1.398186E+01		1.622388E+01		
	!Z_	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	!
		-1-		-2-		-3-		-4-		
-1-		1.993352E+07		2.093667E+07		2.193981E+07		2.294295E+07		
	!Y_	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	* * * *	!
		-1-		-2-		-3-		-4-		
-1-		-1.035022E+02		-1.302175E+02		-1.569329E+02		-1.836482E+02		
	!Q_	* * * *							!P_	* * * *
		-1.200000E+00								-3.140000E+00

As another example, consider the mathematical problem of solving a system of equations

$$Au = v$$

where A is assumed to be a nonsingular matrix of order n; u and v are vectors each having n components. Mathematically we can solve this system, under appropriate conditions, by choosing an initial approximation z^0 and computing the sequence

$$z^{k+1} = Bz^k + w \quad \text{for } k = 0, 1, \dots$$

where $B = I - sA$, $w = sv$, and s is a specified constant. Since this algorithm is merely a form of successive approximations, we know that the sequence of vectors $\{z^k\}$ under appropriate conditions, converges to the solution u. Even though this algorithm is not practical, it still illustrates how sequences can be used in OL/2. More practical algorithms will be presented later.

ALGORITHM: MAIN PROCEDURE;

N = 4;

LET A AND B BE MATRICES OF ORDER (N), AND I THE
IDENTITY MATRIX OF ORDER (N); LET u, v, AND w BE
VECTORS OF ORDER (N), AND c AND s SCALARS; LET z[k]
BE A SEQUENCE MOD(2) OF VECTORS OF ORDER (N);

INPUT A, v; OUTPUT A, v;
s = 2/2.6; B = I - s*A; w = s*v;
c = 0.0001; m = 20;

z[0] = 0; z[1] = c;

FOR k = 0, 1, ... ,m OR UNTIL ||z[k+1] - z[k]|| < c;
COMPUTE:

z[k+1] = B*z[k] + w;

END;

SOLUTION:

OUTPUT z[k]' ;

RESIDUAL_VECTOR:

OUTPUT (A*z[k] - v)' ;

END ALGORITHM;

	!A	*	*	*	*	*	!		!V	!
	-1-	-2-	-3-	-4-					-1-	
-1-	1.00	0.42	0.54	0.66				-1-	0.30	
-2-	0.42	1.00	0.32	0.44				-2-	0.50	
-3-	0.54	0.32	1.00	0.22				-3-	0.70	
-4-	0.66	0.44	0.22	1.00				-4-	0.90	

	!Z	!	21	!	*	*	*	*	*	*	*	*	*	*	*	!
	-1-	-1-	-2-	-3-	-4-											
-1-	-1.235599E+00	4.769411E-02	1.029643E+00	1.468001E+00												
	!(A*Z	!	21	!	-V	!	* <td>* <td>* <td>* <td>* <td>* <td>* <td>* <td>* <td>!</td> </td></td></td></td></td></td></td></td>	* <td>* <td>* <td>* <td>* <td>* <td>* <td>* <td>!</td> </td></td></td></td></td></td></td>	* <td>* <td>* <td>* <td>* <td>* <td>* <td>!</td> </td></td></td></td></td></td>	* <td>* <td>* <td>* <td>* <td>* <td>!</td> </td></td></td></td></td>	* <td>* <td>* <td>* <td>* <td>!</td> </td></td></td></td>	* <td>* <td>* <td>* <td>!</td> </td></td></td>	* <td>* <td>* <td>!</td> </td></td>	* <td>* <td>!</td> </td>	* <td>!</td>	!
	-1-	-1-	-2-	-3-	-4-											
-1-	9.320908E-03	4.148950E-03	6.421390E-04	1.288973E-05												

SUBARRAYS: MAIN PROCEDURE;

N = 6;

LET A BE A LOWER TRIANGULAR MATRIX OF ORDER (N);
INPUT A; OUTPUT A;

FOR K=3,4,...,N-1; PARTITION A AFTER ROWS K-1, K;
SET M=A<3,1>, C=A<3,2>;

OUTPUT M, C;

END;

END SUBARRAYS;

	!A	*	*	*	*	*	!
	-1-	-2-	-3-	-4-	-5-	-6-	
-1-	1						
-2-	2	3					
-3-	4	5	6				
-4-	7	8	9	10			
-5-	11	12	13	14	15		
-6-	16	17	18	19	20	21	

	!M	*	!		!C	!
	-1-	-2-			-1-	
-1-	7	8		-1-	9	
-2-	11	12		-2-	13	
-3-	16	17		-3-	18	

	!M	*	*	!		!C	!
	-1-	-2-	-3-			-1-	
-1-	11	12	13		-1-	14	
-2-	16	17	18		-2-	19	

	!M	*	*	*	!		!C	!
	-1-	-2-	-3-	-4-			-1-	
-1-	16	17	18	19		-1-	20	

DECOMPOSITION: MAIN PROCEDURE;

N = 4;

LET A BE A LOWER TRIANGULAR MATRIX OF ORDER (N);
SET L=A; INPUT A; OUTPUT A;

FOR K=1,2,...,N; PARTITION A AFTER ROWS K-1, K;
SET R=A<2,1> ROW VECTOR, D=A<2,2> SCALAR, C=A<3,2>
COLUMN VECTOR, AND M=A<3,1>;

D = SQRT(D - (R',R'));

C = (C - M×R')/D;

END;

OUTPUT L | L×L';

END DECOMPOSITION;

```
!A * * * * * * * * * * * * !
      -1-      -2-      -3-      -4-
-1-  5.000000E+00
-2-  7.000000E+00  1.000000E+01
-3-  6.000000E+00  8.000000E+00  1.000000E+01
-4-  5.000000E+00  7.000000E+00  9.000000E+00  1.000000E+01

!L * * * * * * * * * * * * !
      -1-      -2-      -3-      -4-
-1-  2.236068E+00
-2-  3.130495E+00  4.472136E-01
-3-  2.683282E+00 -8.944272E-01  1.414214E+00
-4-  2.236068E+00  3.972055E-15  2.121320E+00  7.071068E-01

!L*L' * * * * * * * * * * * * !
      -1-      -2-      -3-      -4-
-1-  5.000000E+00  7.000000E+00  6.000000E+00  5.000000E+00
-2-  7.000000E+00  1.000000E+01  8.000000E+00  7.000000E+00
-3-  6.000000E+00  8.000000E+00  1.000000E+01  9.000000E+00
-4-  5.000000E+00  7.000000E+00  9.000000E+00  1.000000E+01
```

REFERENCES

- Jurich, Dale R. "An Approach to the Compilation of Array Expressions in the OL/2 Language." Department of Computer Science, University of Illinois at Champaign-Urbana, Report No. UIUCDCS-R-72-550.
- Phillips, J. Richard. "The Structure and Design Philosophy of OL/2 - An Array Language - Part I: Language Overview" (to appear).
- Phillips, J. Richard. "The Structure and Design Philosophy of OL/2 - An Array Language - Part II: Algorithms for Dynamic Partitioning." Department of Computer Science, University of Illinois at Champaign-Urbana. Report No. UIUCDCS-R-71-420.
- Phillips, J. Richard and Adams, H. C. "Dynamic Partitioning for Array Languages." CACM, December, 1972.

[This research was supported in part by the National Science Foundation under National Science Foundation Grant GJ-328.]

J. Richard Phillips
Dale Jurich
Robert Bloemer
Eric Tangman

11. GENERAL DEPARTMENT INFORMATION

11.1 Personnel

The number of people associated with the Department in various capacities is given in the following table:

	<u>Full-time</u>	<u>Part-time</u>	<u>FTE</u>
Faculty	22	5	24.46
Visiting Faculty	1	2	2.50
Research Associates and Instructors	---	---	----
Graduate Research Assistants	1	69	35.25
Graduate Teaching Assistants	0	3	15.75
Professional Personnel	2	1	2.50
Administrative and Clerical	17	0	17.00
Nonacademic Personnel (Monthly)	17	0	17.00
Nonacademic Personnel (Hourly)	0	53	13.63
TOTAL.....	60	162	127.59

The Department Advisory Committee consists of Professor J. N. Snyder, Head of Department, Professors E. K. Bowdon, D. F. Cudia, K. W. Dickman, M. F. Faiman, H. G. Friedman, C. W. Gear, D. B. Gillies, W. J. Kubitz, D. J. Kuck, C. L. Lui, R. G. Montanelli, S. Muroga, T. A. Murrell, J. Nievergelt, J. R. Phillips, W. J. Poppelbaum, S. R. Ray, E. M. Reingold, J. E. Robertson, P. E. Saylor, D. L. Slotnick, J. E. Vander Mey, D. S. Watanabe, and T. Wilcox.

11.2 Bibliography

During the third quarter, the following publications were issued by the laboratory:

Report Numbers

- | | | |
|-----|----|--|
| No. | 16 | Bezdek, "Economic Research Group Working Paper No. 6: Simulating The Employment Impacts of the Urban Coalition's Counterbudget," CAC Document No. 16, University of Illinois at Urbana-Champaign, October 15, 1971. |
| No. | 30 | Denenberg, S. A., "A 10-Page Description of the ILLIAC IV System," CAC Document No. 30, University of Illinois at Urbana-Champaign, October 12, 1971. |
| No. | 31 | Grothe, D. M., "PEESPOL Overview," CAC Document No. 31, Univeristy of Illinois at Urbana-Champaign, July, 1971. |
| No. | 34 | McDaniel, L. M., "Symmetric Decomposition of Positive Definite Band Matrices and the Corresponding Solution of Systems of Linear Equations on ILLIAC IV," CAC Document No. 34, University of Illinois at Urbana-Champaign, July 1, 1972. |
| No. | 40 | Pace, S. A., "An ILLIAC IV Gaussian Elimination," CAC Document No. 40, University of Illinois at Urbana-Champaign, September 1, 1972. |
| No. | 41 | Miura, K., "The Block-Iterative Method for ILLIAC IV," CAC Document No. 41, University of Illinois at Urbana-Champaign, May 10, 1972. |
| No. | 42 | Denenberg, S. A., "Getting Started on the ARPA Network Terminal System (ANTS)," CAC Document No. 42, University of Illinois at Urbana-Champaign, August 1, 1972. |
| No. | 43 | Ray, R. M., "BLOCKS: A Fortran IV Program for Plotting Planar Projections of Three-Dimensional Block Models," CAC Document No. 43, University of Illinois at Urbana-Champaign, June 15, 1972. |
| No. | 44 | Hopkin, D. J., "Information Retrieval and the ILLIAC IV," CAC Document No. 44, University of Illinois at Urbana-Champaign, May, 1972. |
| No. | 46 | Lermit, R. J., "A Linear Programming Implementation," CAC Document No. 46, University of Illinois at Urbana-Champaign, October 1, 1972. |
| No. | 47 | Bouknight, W. J., "ANTS - A New Approach to Accessing the ARPA Network," CAC Document No. 47, University of Illinois at Urbana-Champaign, July 1, 1972. |

Reports Cont'd

- No. 521 Bracha, Amnon, "A Method for Solving Polynomial Equations by Continued Fractions," July, 1972.
- No. 531 Jayaramamurthy, S. N. and McCormick, B. H., "Analysis of Texture," July, 1972.
- No. 539 Bond, William D. and Davidson, Edward S., "Realization of the WEB Language," August, 1972.
- No. 540 Ercegovac, Milos D., "RADIX 16 Evaluation of Some Elementary Functions," August, 1972.
- No. 542 Robertson, James E. and Trivedi, Kishor, "The Status of Investigations into the Use of Continued Fractions for Computer Hardware," August, 1972.
- No. 547 Davis, Al; Hart, Jim; Oxley, Don, and Phillips, Keith, "ETS Student's Guide," July, 1972.

Theses

- No. 526 Tu, Kai-Wen, "Stability and Convergence of General Multistep and Multivalued Methods with Variable Step Size," July, 1972 (Ph.D. Thesis).
- No. 534 Polley, Eugene J., Jr., "An Assembler for Efficient File Manipulation," August, 1972 (M.S. Thesis).
- No. 535 Estelita, Thomas A., "Educational Projects for a Digital Logic Laboratory," July, 1972 (M.S. Thesis).
- No. 536 Miller, James Douglas, "The Design and Implementation of a Multi-task Batch Operating System and Paging Interpreter for the PDP-11," August, 1972 (M.S. Thesis).
- No. 537 Fitzgerald, James T., "Load Regulation and Dispatching in a Network of Computers," August, 1972 (M.S. Thesis).
- No. 538 Barr, William J., "Cost Effective Analysis of Network Computers," August, 1972 (M.S. Thesis).
- No. 541 Ercegovac, Milos D., "RADIX 16 Division, Multiplication, Logarithmic and Exponential Algorithms Based on Continued Product Representations," August, 1972 (M.S. Thesis).
- No. 550 Jurich, Dale, "An Approach to the Compilation of Array Expressions in the OL/2 Language," September, 1972 (M.S. Thesis).

Theses Cont'd

Ericksen, J. H., "A Survey of Iterative Methods for Solving Poisson's Equation and Their Adaptability to ILLIAC IV," June, 1972 (Ph.D. Thesis).

Lermit, R. J., "Numerical Methods for the Identification of Differential Equations," June, 1972 (Ph.D. Thesis).

Wilhelmson, R. B., "The Numerical Simulation of a Thunderstorm Cell in Two- and Three-Dimensions," June, 1972 (Ph.D. Thesis).

11.3 Abstracts

E. K. Bowdon, Sr., "Networks Computer Modeling," Processing of the ACM, August, 1972, pp. 254-264.

Abstract:

Network computers are becoming a reality in the seventies. While systems such as the ARPA network, Carnegie Mellon's PLN, the Collins C-System, CDC's Cybertnet, and GE's Time Share Net are coming to fruition, even more grandiose systems are being discussed. These networks all offer the designer the potential of combining the advantages of data base sharing, resource sharing, and load leveling with those of message switching. From a postulation of the essential characteristics of a network computer currently under development, we formulate a queueing theory model for a multi-server system with a finite length priority queue. Then, under the assumptions of Poisson input and exponentially distributed processing times, we use this idealized mathematical model to investigate the steady state stochastic behavior of jobs in the network. We are particularly interested in the efficiency of computer utilization and the average waiting time for jobs of different priority classes.

W. J. Barr, "Cost Effective Analysis of Network Computers," Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, Report No. UIUCDCS-R-72-538, August, 1972.

Abstract:

With the advent of network computers, a new area of computer systems analysis has evolved. Unfortunately, most of the work to date merely extends the previously existing theory of communications. Our analysis is predicated on the assumption that a geographically distributed network computer is quite different from telephone networks and individual computing centers. The potential advantages, from a businessman's point of view, of network computers are discussed and some of the more important problems which arise are presented. We take a look at several existing networks and examine their goals and solutions to these kinds of problems. We develop a priority assignment technique for the individual centers that comprise the network and determine load leveling rules for the entire network. In addition, we analyze this system using classical queueing theory techniques.

R. H. Bezdek, "Economic Research Group Working Paper No. 6: Simulating The Employment Impacts of the Urban Coalition's Counterbudget," CAC Document No. 16, University of Illinois at Urbana-Champaign, October 15, 1971.

Abstract:

This paper describes a large scale computer simulation conducted to determine the detailed manpower effects which may be generated by implementing the Urban Coalition's recommended reorderings of national priorities and Federal expenditures in the period 1972-1976. The procedures followed in each stage of the analysis are

J. T. Fitzgerald, "Load Regulations and Dispatching in a Network of Computers," Department of Computer Science, University of Illinois at Urbana-Champaign. Urbana, Illinois, Report No. UIUCDCS-R-72-537, August, 1972.

Abstract:

This paper is aimed at developing tools to control efficiently the flow of jobs and job traffic in a network of computers. Input of jobs to each center is controlled by predetermined information based on probabilities and stored in table form. These probabilities are developed mathematically, predicated on the fact that we consider the input rate to be a random variable capable of assuming any size. The table is then extended to handle the dispatching of jobs that must be rerouted between different centers in the network and an efficient controller is thus developed.

D. E. Gold, "Applications of Some Switching Network Results to Dynamic Allocation of Memories in a Hierarchy," Compcon 72, IEEE Computer Society Conference, San Francisco, Sept. 1972.

Abstract:

A class of rearrangeable switching networks contains results which are directly applicable to the configuration and use of a computer system memory hierarchy. This paper discusses a decomposition of such networks and relates this to the retrieving of data in a memory hierarchy. Some results of this procedure are summarized and a generalization to arbitrarily many memory levels is explained.

D. M. Grothe, "PEESPOL Overview," CAC Document No. 31, University of Illinois at Urbana-Champaign, July, 1971.

Abstract:

PEESPOL, an acronym for PDP Eleven Executive System Programming Oriented Language, was designed and implemented at the University of Illinois for the specific application of writing an ARPA Net Terminal System for the PDP-11. The general idea was to create a higher level implementation language for the PDP-11. This decision was doubtless influenced by the presence of much experience with Burroughs B5500 and B6500 systems, both of whose operating systems are written in a higher level implementation language. PEESPOL "looks like" ALGOL. That is, it is a block structured language, storage is allocated via declarations, etc. PEESPOL also contains PAL Assembly Language statements and a rather powerful macro generator. PEESPOL itself is a Burrough B6500 ALGOL program (a version also exists for the B5500). The compiler accepts input in card-image format and creates a disk file containing the object code. No special loader is required to load the object code into the PDP-11, the code file being a byte-image of a PDP-11 absolute loader input tape.

specified in detail. The economic model employed is an expanded open input-output model transformed into labor units and integrated with industrial and occupational manpower data. Simulations are run to determine the differential employment impacts which would be caused by the Urban Coalition's budget priorities as opposed to the anticipated expenditure distributions being forecast for the near future. The findings indicate that the Urban Coalition's Counterbudget may cause such drastic dislocations in the labor market that its implementation may be infeasible.

W. J. Bouknight, "ANPS - A New Approach to Accessing the ARPA Network," CAC Document No. 47, University of Illinois at Urbana-Champaign, July 1, 1972.

Abstract:

This brochure describes a new system for accessing the ARPA Network. Providing economical access with terminals and a wide assortment of peripheral hardware, the ARPA Network Terminal System is an access answer for network sites which do not have their own in-house large-scale computer system.

_____, "The ILLIAC IV System," Proceedings of the IEEE, Vol. 60, No. 4, pp. 369-388, April, 1972.

Abstract:

The reasons for the creation of Illiac IV are described and the history of the Illiac IV project is recounted. The architecture or hardware structure of the Illiac IV is discussed--the Illiac IV array is an array processor with a specialized control unit (CU) that can be viewed as a small stand-alone computer. The Illiac IV software strategy is described in terms of current user habits and needs. Brief descriptions are given of the systems software itself, its history, and the major lessons learned during its development. Some ideas for future development are suggested. Applications of Illiac IV are discussed in terms of evaluating the function $f(x)$ simultaneously on up to 64 distinct argument sets x . Many of the time-consuming problems in scientific computation involve repeated evaluation of the same function on different argument sets. The argument sets which compose the problem data base must be structured in such a fashion that they can be distributed among 64 separate memories. Two matrix applications: Jacobi's algorithm for finding the eigenvalues and eigenvectors of real symmetric matrices, and reducing a real nonsymmetric matrix to the upper-Hessenberg form using Householder's transformations are discussed in detail. The ARPA network, a highly sophisticated and wide ranging experiment in the remote access and sharing of computer resources, is briefly described and its current status discussed. Many researchers located about the country who will use Illiac IV in solving problems will do so via the network. The various systems, hardware, and procedures they will use is discussed.

E. W. Davis, Jr., "Concurrent Processing of Conditional Jump Trees." Compton 72. IEEE Computer Society Conference, San Francisco. Sept. 1972.

Abstract:

Software algorithms and a hardware decision processor are presented for speeding up the execution of IF statement trees. The techniques used, designed for a multiprocessor environment, increase the execution concurrency.

S. A. Denenberg, "A 10-Page Description of the ILLIAC IV System," CAC Document No. 30, University of Illinois at Urbana-Champaign, October 12, 1971.

Abstract:

The architecture or hardware structure of the ILLIAC IV System is discussed. The ILLIAC IV Array is a Vector or Array Processor with a specialized Control Unit that can be viewed as a small stand-alone computer by itself. The text has been revised and condensed from ILLIAC IV Document No. 225.

, "Getting Started on the ARPA Network Terminal System (ANTS)," CAC Document No. 42, University of Illinois at Urbana-Champaign, August 1, 1972.

Abstract:

Written specifically for a non-programmer computer user, this document is a tutorial on the use of the ARPA Network through the PDP-11 "ANTS" system at the Center for Advanced Computation. Topics covered include how to turn on and operate the two available types of remote terminal at the Center (the Hazeltine 2000, and Texas Instruments' "Silent 700"), the theory of operation of the ARPA Network, the minimum set of ANTS commands to allow the user to begin exploring the ARPA Network, and a heavily annotated example of one specific use of the Network -- writing a program in the BASIC computer language, and executing it on the PDP-10 at the University of Southern California Information Sciences Institute in Los Angeles, California. The program would be run from a remote terminal located at the Center for Advanced Computation.

J. H. Erickson, "A Survey of Iterative Methods for Solving Poisson's Equation and Their Adaptability to ILLIAC IV," Ph.D. Thesis directed by D. L. Slotnick, June, 1972.

Abstract:

This study examines the relative utility and efficiency of several iterative schemes (SOR, SLOR, ADI) for solving the Poisson equation ($\nabla^2 u = f$) using the ILLIAC IV computer. High level language (FORTRAN) codes are given for each scheme and timing estimates are provided for the kernel of each scheme written in assembly (ASSEMBLY) code.

D. J. Hopkins, "Information Retrieval and the ILLIAC IV," CAC Document No. 44, University of Illinois at Urbana-Champaign, May, 1972.

Abstract:

The use of the ILLIAC IV for information retrieval has been investigated. This paper recapitulates much of the work done by Barry A. Wittman in "An Investigation Into Information Retrieval Utilizing ILLIAC IV," CAC Document No. 22, and offers some further reasons for concluding that the ILLIAC IV should not be used as an information retrieval machine.

D. R. Jurich, "An Approach to the Compilation of Array Expressions in the OL/2 Language." Department of Computer Science, University of Illinois at Urbana-Champaign Urbana, Illinois, Report No. UIUCDCS-R-72-550.

Abstract:

This report is concerned with the compilation of array expressions, such as appear in the language OL/2. Operator Language 2. The basic operation and operand types that may appear in an array expression in OL/2 are defined, as well as the data structures used to represent them. The parsing of source expressions into an intermediate expression tree form is described, and an algorithm for generating object code from the intermediate representation is given. The problem of reducing, through compilation techniques, the storage necessary for run time temporary results is discussed. The array expression compilation module for OL/2, as implemented, is presented in the Appendix.

D. J. Kuck, D. H. Lawrie, and Y. Muraoka, "Interconnection Networks for Processors and Memories in Large Systems," Compeon 72. IEEE Computer Society Conference. San Francisco Sept. 1972.

Abstract:

In computers which use a large number of independent memories and/or processors, the interconnection of these units becomes a serious problem. We outline some of the machine and program requirements and discuss the relations between them.

D. J. Kuck and Y. Muraoka, "Fast Computers from Slow Parts," Compeon 72 IEEE Computer Society Conference, San Francisco. Sept. 1972.

Abstract:

The analysis of ordinary FORTRAN programs indicates that a number of operations may be performed simultaneously, increasing effective speed of machines without using faster parts. Several design matters are sketched including accessing, aligning and processing data.

R. J. Lermitt, "A Linear Programming Implementation," CAC Document No. 46,
University of Illinois at Urbana-Champaign, October 1, 1972.

Abstract:

This document discusses the implementation of a recently developed version of the simplex method for linear programming using Cholesky factorization (where the basis is expressed as a product of a lower triangular and an orthogonal matrix) rather than the more usual product form of the inverse. Storage schemes for the resulting lower triangular matrix suitable for an array processor (the orthogonal matrix is not retained) and necessary adaptations to the algorithm when the constraint matrix is expressed as a linear operator are also discussed.

R. J. Lermitt, "Numerical Methods for the Identification of Differential Equations," Ph.D. Thesis directed by D. L. Slotnick, June, 1972.

Abstract:

This dissertation considers computational methods designed to aid in mathematical model building. Specifically, it discusses methods of determining ordinary differential equations given their solution in the form of observed data.

Since the problem cannot be solved in this generality, it is necessary to supply equations containing arbitrary functions. The problem is then to find these functions given the solution of the equations. In order to be amenable to computer solution, a discretization of the functions as a linear sum of a given orthonormal set, is necessary. The problem thus reduces to one of finding a finite number of parameters.

The solution technique is to find that function which produces a solution most closely approximating the observed data. It is thus a problem in the minimization of nonlinear functionals and may be solved by iterative methods. Modifications required to ensure that the functional is convex, thus guaranteeing a global minimum solution, are discussed. Different algorithms considered for carrying out the minimization include the generalization of Newton's method and variants of it which are more economical in computer time, especially the Conjugate Gradient method. All of these methods require derivatives which are derived automatically from the original equation using formal algebraic manipulation. The different methods are compared for rates of convergence and amount of calculation required at each iteration.

Two examples are included. The effect of introducing random errors into the data to simulate observational errors, and how this may alter the convergence rates, is also discussed.

L. M. McDaniel, "Symmetric Decomposition of Positive Definite Band Matrices and the Corresponding Solution of Systems of Linear Equations on ILLIAC IV," CAC Document No. 34, University of Illinois at Urbana-Champaign, July 1, 1972.

Abstract:

An algorithm to solve systems of linear equations for many right hand sides $Ax = b$ where A is a symmetric positive definite banded matrix is described for implementation on ILLIAC IV. Two codes are discussed -- one for band width less than 64, and one for band widths ranging between 64 and 127.

K. Miura, "The Block-Iterative Method for ILLIAC IV," CAC Document No. 41, University of Illinois at Urbana-Champaign, May 10, 1972.

Abstract:

The Block-Iterative Method for solving elliptic partial differential equations is discussed. This method proved to be quite suitable for parallel computation. The parallel algorithm and storage scheme on the ILLIAC IV are described, and GLYPNIR programs are also attached.

S. A. Pace, "An ILLIAC IV Gaussian Elimination," CAC Document No. 40, University of Illinois at Urbana-Champaign, September 1, 1972.

Abstract:

Gaussian Elimination is a method used to find the solution vector \underline{x} of the equation $A\underline{x} = \underline{b}$ where A is any general, non-singular, square matrix and \underline{b} is a vector. The process can be broken down into two independent phases:

- (1) The triangular decomposition of the matrix A , and
- (2) The back substitution process to find the solution \underline{x} .

This paper describes the Gaussian Elimination algorithm and the ILLIAC IV routines which perform the process.

J. Richard Phillips, "The Structure and Design Philosophy of OL/2 - An Array Language - Part II: Algorithms for Dynamic Partitioning," Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDSC-71-420.

Abstract:

The classical process of partitioning an array into subarrays is extended to a more useful array language operation. Various modes of partitioning are defined for different types of arrays, so that subarrays may vary over the original array in a nearly arbitrary manner. These definitions are motivated with several realistic examples to illustrate the value of partitioning for array languages.

Of general interest in the data structure for partitioning. This consists of dynamic tree structures which are used to derive and maintain the array control information. These are described in sufficient description presented in this paper is implemented in a new array language, OL/2, currently under development at the University of Illinois.

R. M. Ray, "BLOCKS: A Fortran IV Program for Plotting Planar Projections of Three-Dimensional Block Models," CAC Document No. 43, University of Illinois at Urbana-Champaign, June 15, 1972.

Abstract:

This report explains the use of BLOCKS: a Fortran IV program for plotting planar projections of three-dimensional block compositions. We define a block composition as any physical object, scale model, or as a spatial composition of straight lines, rectangular planes, and rectangular solids. For any particular view of a composition, the program plots only those lines that should be visible to the observer.

F. R. Salz, "A GPSS Simulation of the 360/75 Under HASP and O.S. 360," Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, Report No. UIUCDCS-R-72-528, June 1972.

Abstract:

The success or failure of a computing system in today's highly competitive market is often determined by the efficiency of its operating system. Consequently, existing operating systems are constantly being modified, extended, and, hopefully, improved. The key question pertaining to the implementation of proposed changes is? "Does the proposed change improve the existing operating system?" One appealing method of answering this question is to simulate the operation of the computing system both under the existing operating system and the system with the proposed changes included. The obvious first step in such a study is to build a model to simulate, with accuracy, the existing system. In this paper, such a model is presented for the IBM 360/75 operating under HASP and O.S. A GPSS simulation of the system is presented and some results are given which verify the accuracy of the simulation.

Kai-Wen Tu, "Stability and Convergence of General Multistep and Multivalued Methods with Variable Step Size," Department of Computer Science Report No. 526, University of Illinois, Urbana, Illinois. Ph.D. Thesis.

In this thesis we are concerned with variable step size multistep methods which are generalizations of

$$\sum_{i=0}^k (\alpha_i y_{n-i} + h\beta_i y'_{n-i}) = 0$$

Shiv Prakash Verma, "Perspective Transformer for the Stereomatrix 3-D Display System," Department of Computer Science Report No. 532, University of Illinois, Urbana-Champaign, October, 1972.

Abstract:

STEREOMATRIX is a large screen interactive 3-D display system which presents computer-generated transparent "wire-frame" drawings stereoscopically. A 3-D cross called the cursor can be moved around in the viewing space by the observer by means of a joystick. This feature allows the observer to select a point in the viewing space for graphic manipulation and relay it to the computer space. The perspective of the figure changes with observer movement in such a way that the figure appears to be stationary in space.

R. B. Wilhelmson, "The Numerical Simulation of a Thunderstorm Cell in Two- and Three-Dimensions," Ph.D. Thesis directed by D. L. Slotnick, June, 1972.

Abstract:

A report is given in this paper of the effect of ignoring the environmental pressure deviation when the saturation vapor pressure is required, and this is done by using a two-dimensional deep convection primitive model. The model includes precipitation, and the equations are integrated for the life cycle of a strong cumulus cell. Further, the relationships between the dynamic, buoyancy, and drag pressures, as later defined, are investigated.

A preliminary investigation of the life cycle of a cumulus cell in three-dimensions has been undertaken. Results of the simulation of such a cell that is axisymmetric in nature are compared with a similar two-dimensional slab-symmetric model. The model and computer program are designed for further investigation and expansion.

No serious attempts are made in the following to simulate real clouds in order to compare the results with observational data; however, general comparisons can be and are made. The main purposes are to investigate the importance of the pressure deviation in the condensation process using a two-dimensional model and, then, to make a general comparison between a two- and a three-dimensional cloud simulation.

11.4 Colloquia

"The dynamics of paging," by Professor Maurice V. Wilkes,
Computer Laboratory, University of Cambridge, Corn Exchange
Street, Cambridge CB2 3QG, Massachusetts, September 25, 1972.

11.5 Drafting

During the third quarter, a total of 501 drawings were processed by the general departmental drafting section:

<u>Formal Drawings</u>	
Large Drawings	63
Medium Drawings	152
Small Drawings	156
Layouts	0
Report Drawings	25
Change Order Drawings	50
Miscellaneous Drawings	<u>55</u>
Completed Total Drawings.	501
(M. Goebel)	

11.6 Shop's Production

Job orders processed and completed during the third quarter of 1972 are as follows:

	<u>AEC 2118</u>	<u>AEC 1469</u>	<u>Other</u>
Machine Shop	1	15	1
Electronic Shop	1	87	36
Chemical Shop	2	59	11
Layout Shop	1	43	37
(F. P. Serio)			

DEPARTMENT OF COMPUTER SCIENCE
GRADUATE COLLEGE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

REPORT REQUEST FORM

Report Number

Title

Fold and Staple as shown

NAME:

ADDRESS:

Fill out Mailing Label



NAME:

ADDRESS:

CUT ALONG THIS LINE

STAPLE

FOLD

Mail Room
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

Stamp

FOLD

BIBLIOGRAPHIC DATA HEET	1. Report No. UIUCDCS-QPR-72-3	2.	3. Recipient's Accession No.
Title and Subtitle Quarterly Progress Report			5. Report Date
Author(s)			6.
Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801			8. Performing Organization Rept. No.
Sponsoring Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801			10. Project/Task/Work Unit No.
			11. Contract/Grant No.
			13. Type of Report & Period Covered Quarterly Progress Report July, Aug. - Sept.
Supplementary Notes			14.
Abstracts Not Applicable			
Key Words and Document Analysis. 17a. Descriptors Not Applicable			
b. Identifiers/Open-Ended Terms Not Applicable			
c. COSATI Field/Group			
Availability Statement		19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 158
		20. Security Class (This Page) UNCLASSIFIED	22. Price

QUARTERLY TECHNICAL PROGRESS REPORT

October, November, December 1972



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN - URBANA, ILLINOIS

QUARTERLY TECHNICAL PROGRESS REPORT

October, November, December 1972

UIUCDCS-QPR-72-4

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

TABLE OF CONTENTS

	Page
1. CIRCUIT RESEARCH	1
1.1 Pentecost (Project No. 31)	2
1.2 Ergodic (Project No. 39)	2
1.2.1 Summary	2
1.3 Telemaze (Project No. 41)	2
1.3.1 Project Goal	2
1.3.2 Communication System Progress.	3
1.3.2.1 Encoding Technique	3
1.3.2.2 Decoding Technique	8
1.4 OCOMO (Project No. 42)	8
1.4.1 Summary.	8
1.4.2 Circuit Description.	9
1.5 Holomar (Holographic Matrix Recognizer) (Project No. 44)	16
1.5.1 Introduction	16
1.5.2 Strategy	16
1.6 ASIM (Analog Simulator) (Project No. 45)	19
1.6.1 Introduction	19
1.6.2 A Pure Analog CM	19
1.6.3 A Stochastic CM.	23
1.6.4 A DDA Type CM.	23
1.6.5 A CM Using a Microprocessor.	25
1.6.6 Conclusion	25
1.7 Thespiac (Project No. 47)	28
1.7.1 Functional Characteristics	28
1.7.2 Component Search	32
1.7.3 Summary.	33
2. HARDWARE SYSTEMS RESEARCH.	34
2.1 Lascot (Project No. 09)	35
2.2 LINDA (Project No. 28)	35
2.2.1 Summary.	35
2.2.2 Project Status	36
2.2.2.1 Video Amplifier.	36
2.2.2.2 Double Differentiation.	36
2.2.2.3 Delay Line	36
2.2.2.4 Shade/Erase Logic.	39
2.2.3 Future Work.	39
2.3 Stereomatrix (Project No. 30)	41
2.3.1 Coefficient Generator.	41
2.3.2 Display.	42
2.4 Scantrix (Project No. 35)	42
2.5 Frog (Project No. 36)	42
2.6 Beacon (Project No. 38)	44
2.7 Caecotron (Project No. 43)	44
2.8 CM (Correlation Modulation, Project No. 48).	49
2.8.1 Introduction	49

	Page
2.8.2 The Correlation Modulation System.	50
2.8.2.1 The Delay Line	50
2.8.2.2 The Microphone Amplifier and Delay Line DC Bias..	52
2.8.3 The Correlation Modulation Test Experiment	52
3. SOFTWARE SYSTEMS RESEARCH.	57
3.1 Numerical Processes.	58
3.1.1 DIFSUB	58
3.1.2 SPARSE	59
3.1.3 Remote Data Structure Utilities: COMMUNE.	60
3.1.4 Plot Package	63
3.1.5 Item Analysis.	65
3.1.6 Interpreter Language Assembler . .	73
3.2 Illinois Graphics Computing System (IGCS)..	77
3.2.1 Overlay Supervisor	77
3.2.2 A Drafting Language (ADL).	77
3.2.3 Plot Sectorization Overlay	77
3.2.4 ILLISM-E	77
3.3 Graphical Remote Access Support System (GRASS)..	79
3.3.1 Implementation of GPSS for the Simulation and Modelling System. .	79
3.3.2 Monitors	81
3.3.3 FETCH Routine.	87
4. IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH: ILLIAC III	92
4.1 Semantic Model Interpretation Methods for Image Processing	92
4.2 Covering Theory.	92
4.3 Variable-Valued Logic.	93
4.4 Engineering and Maintenance.	93
5. THEORY OF DIGITAL COMPUTER ARITHMETIC.	96
5.1 Continued Fraction Arithmetic.	96
5.2 Automatic Computation of Certain Elementary Functions Using Microprogramming	96
5.3 Function Evaluation Techniques	96
6. SWITCHING THEORY AND LOGICAL DESIGN.	98
7. MACHINE AND SOFTWARE ORGANIZATION STUDIES.	103

	Page
7.1	FORTRAN Parallelism Detection. 103
7.2	DO Loop Analysis 105
7.3	Memory-Processor Connection Networks 105
7.4	D-Machine Microprogram 105
7.5	Information Retrieval Computers. 106
8.	COMPUTER SYSTEM ANALYSIS 107
8.1	Computer Network Modeling. 107
8.2	Center Throughput Analysis 108
9.	NUMERICAL ANALYSIS 110
10.	COMPUTATIONAL ASPECTS OF COMBINATORIAL ALGORITHMS. . 111
11.	DISTRIBUTED MINI-COMPUTER COMPUTING SYSTEM (MESH GROUP). 112
11.1	MESH Hardware Components 112
11.2	Memory Segmentation Hardware 112
11.3	SUE PDP-11 Emulator. 113
11.4	CRT Display Terminals. 114
11.5	PDP-11 Maintenance 114
11.6	MESH Control/Transfer Mechanism (TM Box) ; . 115
11.7	Microcode Assembler for Lockheed SUE 116
11.8	Lockheed SUE Microcode Simulator 117
11.9	MESH Software. 118
	11.9.1 System Design 118
	11.9.2 System Software 118
12.	GENERAL DEPARTMENT INFORMATION 120
12.1	Personnel. 120
12.2	Bibliography 121
12.3	Abstracts. 123
12.4	Colloquia. 129
12.5	Drafting 130
12.6	Shop's Production. 130

1. CIRCUIT RESEARCH

(Supported in part by the Office of Naval Research under Contract N000 14-67-A-305-0007, W. J. Poppelbaum, Principal Investigator).

Summary

1. Pentecost (Panigrahi)

Completion and assembly of receiver and camera hardware.

2. Ergodic (Cutler)

Final design.

3. Telemaze (Pott)

Method of encoding, transmitting and decoding information from control to remote vehicle.

4. Ocomo (Budzinski)

Functional description of new circuits.

5. Holomar (Huberts)

Holographic Matrix Recognizer - a new project. A set of sensory inputs from an unknown object is compared with a matrix of known signals, subject to appropriate weighting. Outputs allow identification of object's properties (e.g. shape) in probabilistic fashion. Processing is stochastic.

6. Asim (Mudge)

Analog simulator - a new project - aims to employ digital micro-processors in place of analog devices and controllers.

7. Thespiac (Daley)

Design of an intelligent, flexible, digital control system for theatre lighting.

M. Faiman, Editor

1.1 Pentecost (Project No. 31)

In this quarter the hardware design for the Pentecost receiver and camera systems has been completed. The filter wheel with motor and photo-detector have been mounted in front of the camera. The photo-amplifier circuits and the synchronization circuits are mounted near the camera. Power is brought by cables from the receiver.

The high voltage supply was received during this quarter. Other power supplies, the monitor, the penetron tube, switching circuits and other color control circuits have been installed in the receiver cabinet.

G. Panigrahi

1.2 Ergodic (Project No. 39)

1.2.1 Summary

The arithmetic unit of Ergodic is the remaining part to be completed. The designs have been finished for this unit and only debugging of the system is left to be accomplished. Therefore, this report will be the last one appearing in the Quarterly Report. A complete report on Ergodic should be published in the next quarter.

Jim Cutler

1.3 Telemaze (Project No. 41)

1.3.1 Project Goal

The objective of TELEMAZE is to design a simple system with an adjustable feedback delay which could control the trajectory of a distant vehicle. The condition which is characteristic of this system is that the feedback delay time is much greater than the vehicle's movement time.

1.3.2 Communication System Progress

The basic function of this system is to transmit the detected coordinates (from the L.E.D. panel) from an eight bit register to the remote vehicle. Several methods have been investigated which would accomplish this. Before the eight bits of data can be applied to the continuous loop analog recorder (described in the previous report), it must be converted to serial information and encoded. The method chosen uses a shift register and BIPHASE encoding (also called frequency modulation, etc.). This technique was selected because of its simplicity in encoding/decoding and its self-clocking characteristic.

1.3.2.1 Encoding Technique

Figure 1 shows eight bits of information encoded using this method. It is to be noted that each leading edge transition corresponds to a clock (i.e., cell division) and a change of state at the center of the cell period corresponds to a "1" bit.

The logic used to realize this scheme is shown in Figure 2, in block diagram form. New information is loaded into the shift register at a frequency of 250Hz. The data output clock will be 4 kHz. This provides 16 bits which can be transmitted per cycle. Eight of these bits will be data and the remaining bits will be used as flags. The information loaded into the 8-bit shift register will be shifted into the encoder J-K flip-flop. The output J-K flip-flop is allowed to toggle if a 4 kHz clock or a data bit of "1" occurs. This will produce the needed waveform.

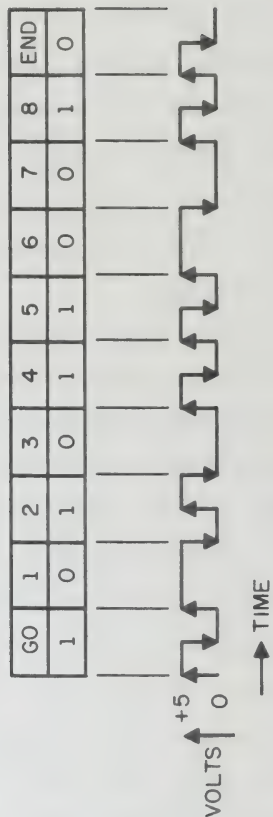


Figure 1. Data Word Format using Biphase Encoding

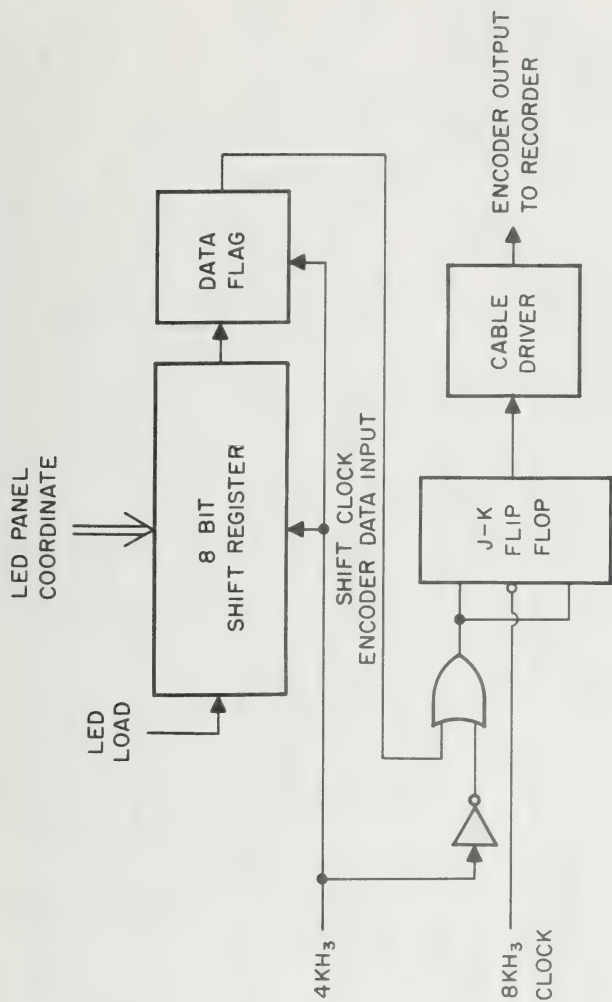


Figure 2. Biphase Encoding Logic (Block Diagram)

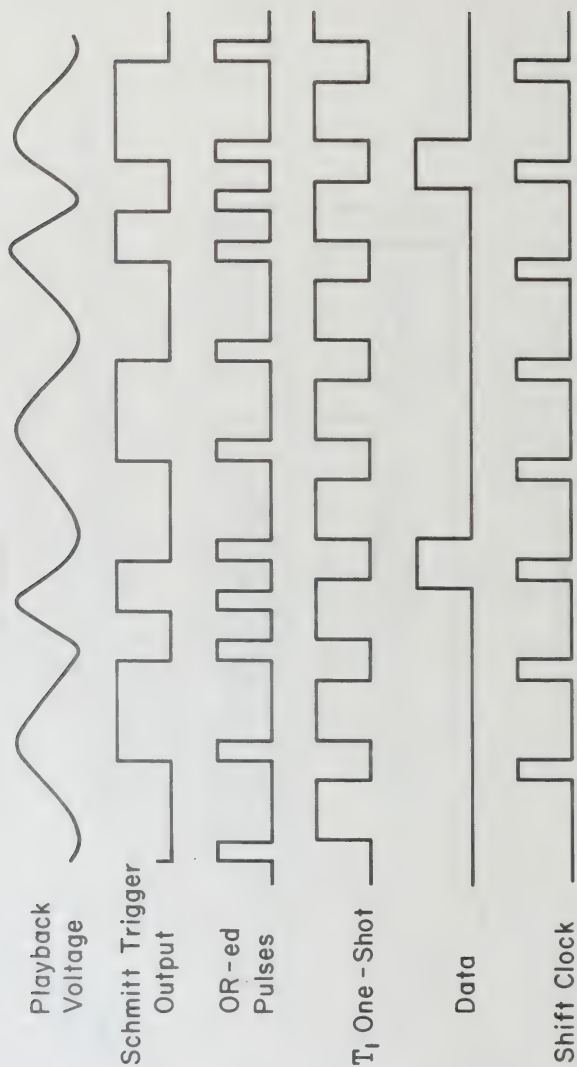


Figure 3. Biphase Decoder Waveforms

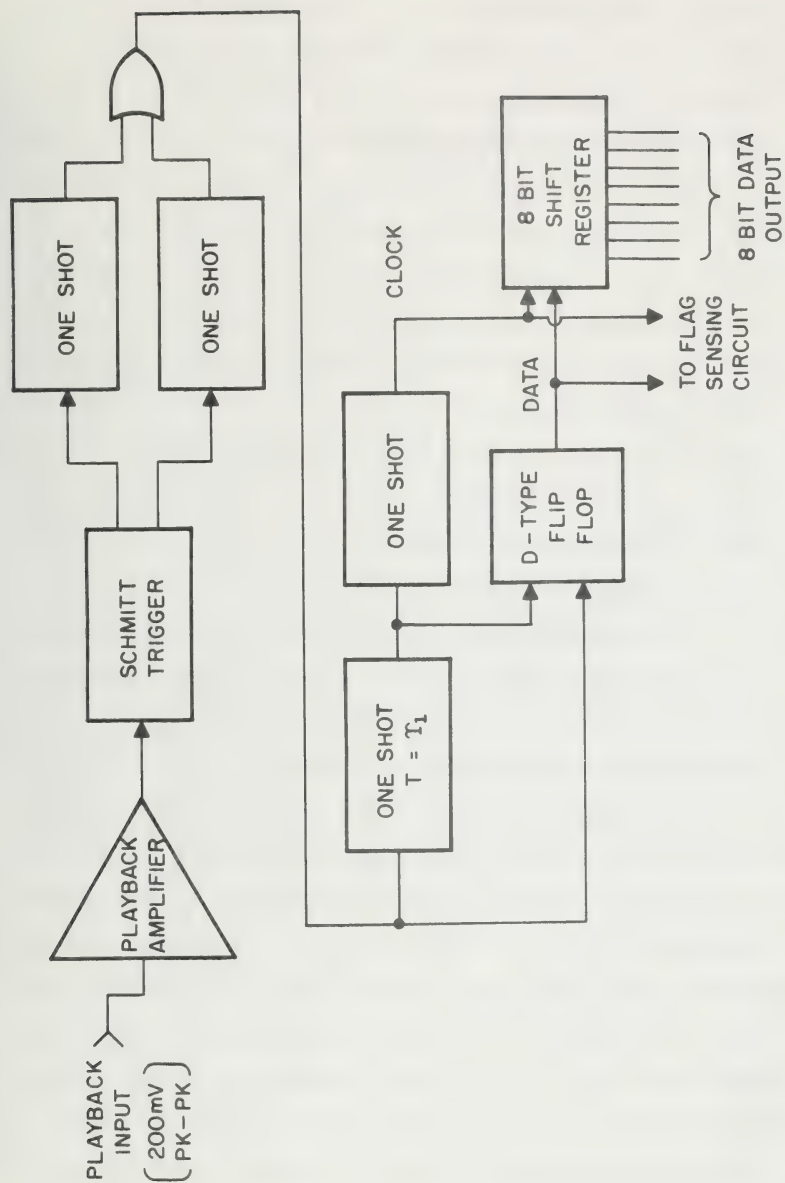


Figure 4. Biphase Decoder Logic (Block Diagram)

1.3.2.2 Decoding Technique

Data received from the continuous loop recorder can easily be decoded into binary information. This is shown in Figure 3. The playback voltage is amplified and applied to a Schmitt trigger. The voltage output of the Schmitt trigger is the reconstruction of the recorded voltage. Then on each transition of the Schmitt trigger a one-shot is fired. These pulses are OR-ed together and applied to circuitry depicted in Figure 4. The original binary data and clock can then be extracted. This data is then converted back to parallel lines with the use of the shift register.

Edward Pott

1.4 OCOMO (Project No. 42)

1.4.1 Summary

During this past quarter, the final circuits for the encoder have been designed and built. Also, a special decoder was designed for testing and maintaining the encoder. With this decoder, the encoder performed well while reproducing voice signals.

In addition to the decoder a photomultiplier circuit for detecting the light encoded voice signal was completed. Also, the vertical deflection circuit was completed, along with a circuit to combine a synchronization pulse for the digital data. (In the previous report, I wrote that the sampled, quantized and stored voice signal would be applied to the vertical plates and a horizontal sweep would be generated. Because of the geometry of the CRT, I have chosen to apply the sampled, quantized and stored voice signal to the horizontal deflection plates and the generated sweep to the vertical deflection plates).

1.4.2 Circuit Description

Figure 1 shows the photomultiplier circuit. The cathode was placed at a high negative voltage because this would put the anode at a level near ground and would, therefore, be easily compatible with integrated circuits. The voltage between the anode and the 9th dynode is about 120 volts. This voltage puts it at current saturation with the light levels of the CRT. The three .005 μ F capacitors serve to maintain the voltage between dynodes constant and thus keep the electron beam focused. The +5V is attached to the 10k load resistor to make the output compatible with TTL, although this feature was not used in the final design.

Figure 2 shows the vertical deflection circuit along with the sync combining circuit. The 74123 is used to generate a clock with 125kHz frequency (8ns period). Three of the bits of the 74191 4 bit binary counter are used to control the output of an MC1406, a D/A convertor. These three bits vary from 0 to 7 in binary. The output of the D/A is converted from current to voltage by a 741C op amp. The waveform is shown at the top of Figure 3. The fourth bit of the counter is used to control the direction of the count. When the counter reaches a maximum or minimum (0 or 15 in binary), pin 12 will go high. On the falling edge of the clock, Q of the JK flip-flop will go to a 1. This inhibits the counter for 1 clock pulse and is the reason for the double length of the output at the extremes in Figure 3. The \bar{Q} output of the flip-flop causes a parallel load of the counter. The effect of this is to change the direction of the count. The store command is generated at the extremes of the vertical deflection waveform. The store command changes the data stored in the sample, quantize and store circuit. The appearance of the store command thus causes a new code word to be selected,

.4V

VERTICAL
DEFLECTION
OUT



STORE
COMMAND



POSSIBLE
OUTPUT
SEQ

SYNC PULSE



CLOCK

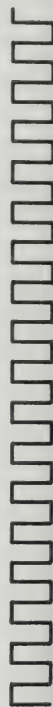


Figure 3. Encoder timing

since the horizontal deflection will be changed so as to place the beam behind a new strip of light and dark spots or rather a new code word strip. I chose this type of vertical waveform because it will have the effect of making a code word associated with one quantized horizontal voltage have two different sequences of 1's and 0's depending on whether the code word was selected when the counter was counting up or down. Of course if the 1st 4 bits of the code word are the mirror image of the last 4, the above effect will be negated.

Because of the nature of this system, the horizontal sweeps of the decoder and encoder will have to be synchronized. I have chosen to combine the sync and data into one signal. The sync pulse is 3 volts above the normal high level of the data. The sync pulse occurs when: the counter has the value 1100, MONOSTABLE A is not triggered, the data level is high, and the Variable Map Select is high.

The Maintenance decoder was designed to test and maintain the encoder. Figure 4 shows the set of strips placed on the front of the CRT. Taking light to be a "0", one can readily see that the strips represent the numbers from 0 to 63 in binary. This uses 6 bits; the remaining 2 vertical bits are always dark. When a horizontal strip has just been selected by the input signal (voice) and the vertical deflection is going from bottom to top, the serial data from the Coded Data and SYNC line will be converted to a parallel form by register A in Figure 5 by a right shift. The direction and phase of the serial data flow are determined by the flip-flop. The flip-flop is initialized by the maintenance decoder sync and changes state after each code word. The 6 bits of useful data are transferred in parallel from register A to register B by the store command. The parallel data of register B is converted to

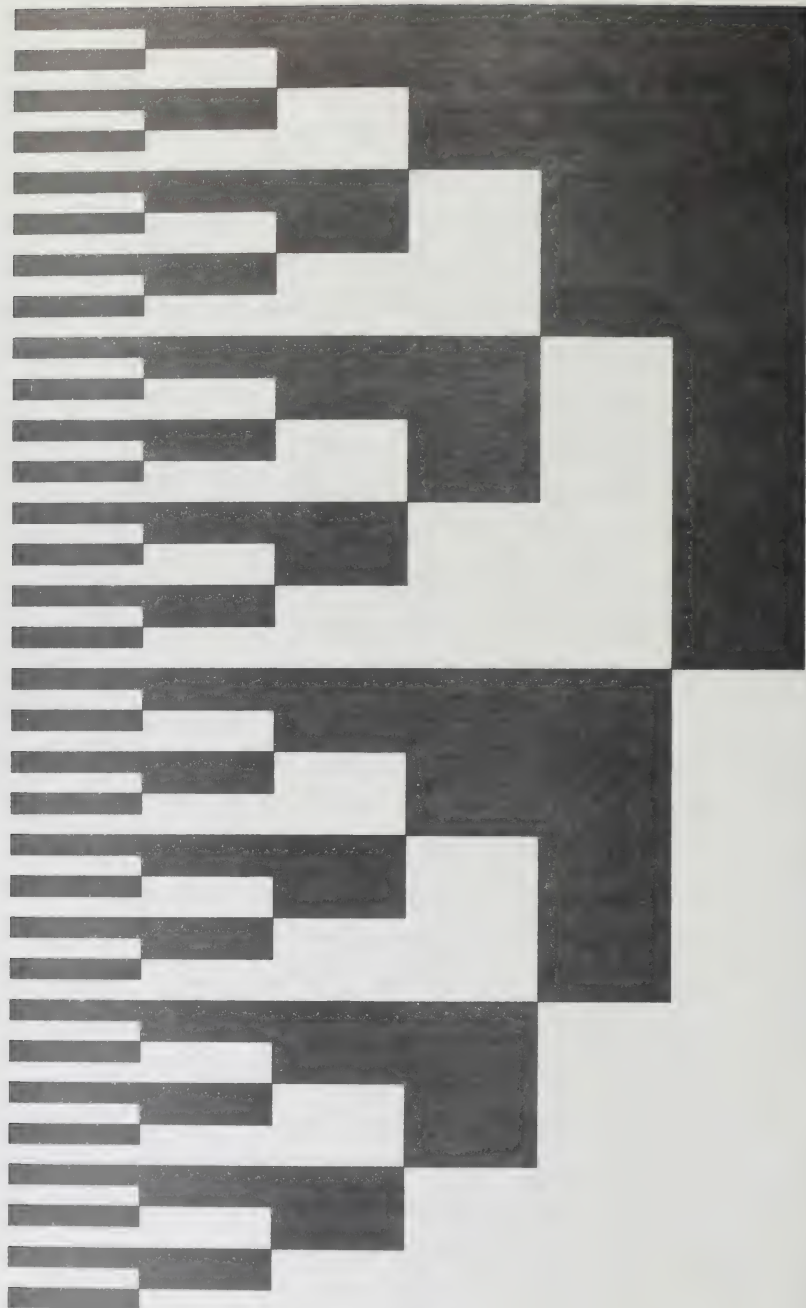


Figure 4. Code Strips

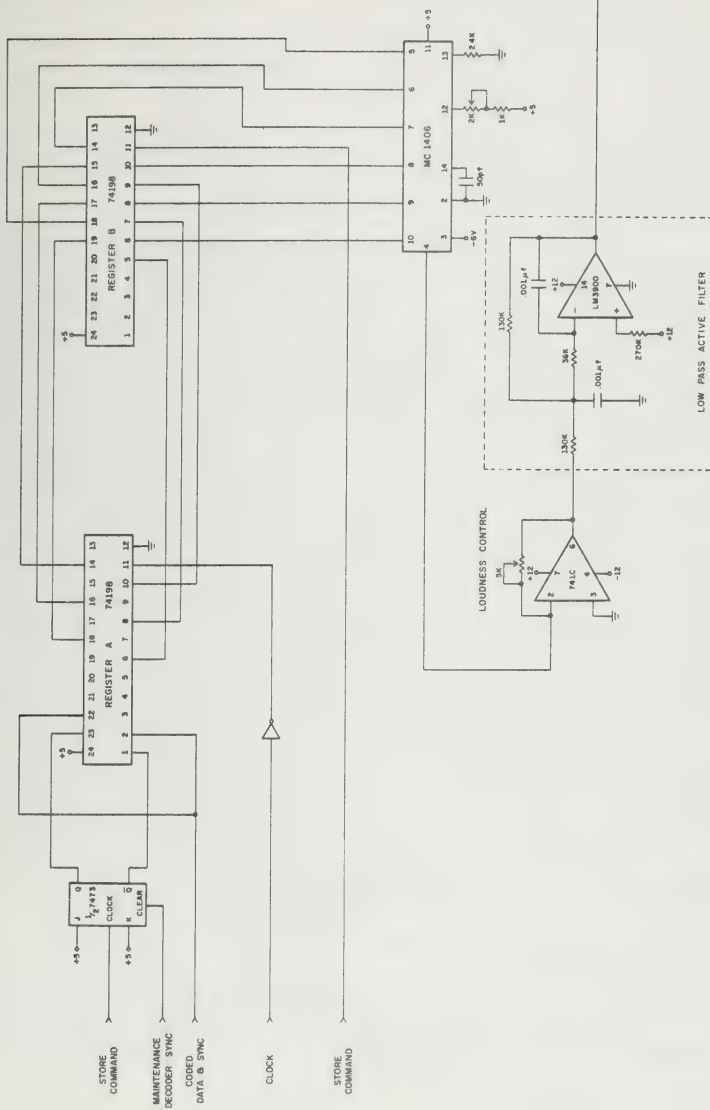


Figure 5. Maintenance Decoder

an analog current by the MC140C D/A. The 74IC op amp provides some gain. The low pass filter has a corner frequency of 7.5kHz with gain and Q of 1. The reproduced signal from the filter is applied to an audio amplifier and speaker.

Robert Budzinski

1.5 Holomar (Holographic Matrix Recognizer) (Project No. 44)

1.5.1 Introduction

Holomar is an engineering system in which sensory inputs (input data in the form of a set of stochastic sequences) are made to interfere with interpretive signals (in the form of a second set of stochastic sequences) in a matrix, the products being stored at the nodes. A catalog of these products can be stored at each node corresponding to different sensory inputs. For recognition of "unknown" sensory inputs, new products are formed. The absolute value of the difference between these new products and previously stored products can then be found for each node and summed over the whole matrix. If this sum is less than a certain threshold (to be set by the operator) recognition has occurred. Clearly this gives arbitrarily fuzzy recognition, and misindication at a few nodes does not upset the system if the threshold is set large enough. There are many ways of implementing this arrangement and cost will be a great factor in deciding exactly how it is done.

1.5.2 Strategy

It was decided that the objective of this machine would be to recognize two-dimensional regular polygons, even under conditions of

rotation, magnification, and translation. It quickly became apparent that the Holomar recognition matrix itself had to be supplied with fairly standardized signals from the input mechanism. Therefore, most of the work done during this quarter has been to design the processing unit that "sees" the polygon and provides the input signals to the Holomar matrix. The input is via a matrix of 487 phototransistors. This matrix is actually a combination of twelve basic matrices, each of which is made up of nine lines (called scan lines) with nine phototransistors per line. Current is summed from all nine phototransistors in a scan line by an op amp whose output voltage is therefore a direct indication of the number of phototransistors receiving light. This gives a measure of the width of an opaque object placed over the matrix as measured along the scan line. For a resolution of ten degrees (a basic matrix for every ten degrees of rotation), twelve basic matrices or 120 degrees of rotation are needed to see all the views of a triangle, the least sided equilateral polygon. The pattern of the basic matrix was made hexagonal so that a great deal of sharing of phototransistors could occur between a basic matrix and the one following it by sixty degrees (see Figure 1). The overall input matrix then appears as a series of concentric circles about a center point which is the fifth phototransistor of the fifth scan line of each of the twelve basic matrices.

The total output, then, appears as voltages at the outputs of 108 (nine scan lines by twelve basic matrices) op amps. The signals must then be standardized to take care of rotation and magnification of the object. Different methods of accomplishing this are now being studied.

Garlan Huberts

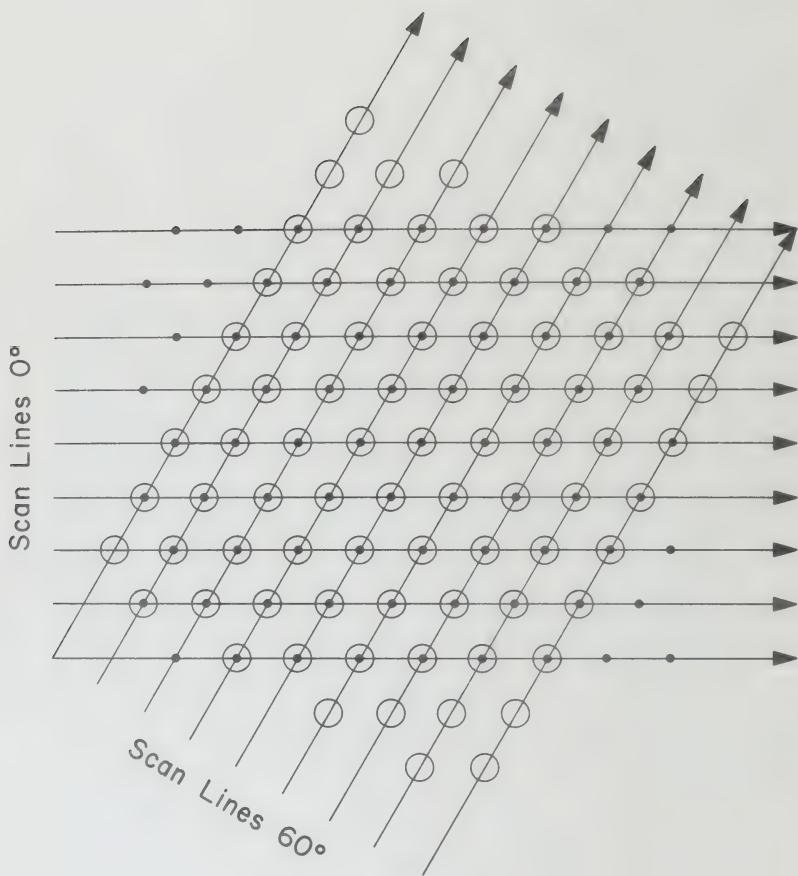


Figure 1. Sharing of Points Between Two Basic Matrices

1.6 ASIM (Analog Simulator) (Project No. 45)

1.6.1 Introduction

The aim of this project is to develop an "all-purpose box," which can be used in place of common analog computing elements, such as summers, differentiators, integrators, multipliers, etc. Such an element could be used as a basic building block in:

1. Constructing an analog chain, from a set of transducers to a set of recording instruments.
2. Solving a differential equation.
3. Constructing models of systems and processes, that could be used as on-line reference models.
4. In-flight computing.

In order that the elements are easy to use and that rapid programming is possible with little experience, a universal computing module (CM), has been proposed, which can furnish the following functions:

1. addition,
2. subtraction,
3. multiplication,
4. division,
5. integration,
6. differentiation.

A design study was carried out to determine the best way to implement the CM. Four general methods of implementation were evaluated.

1.6.2 A Pure Analog CM

A schematic for a pure analog CM is shown in Figure 1. This

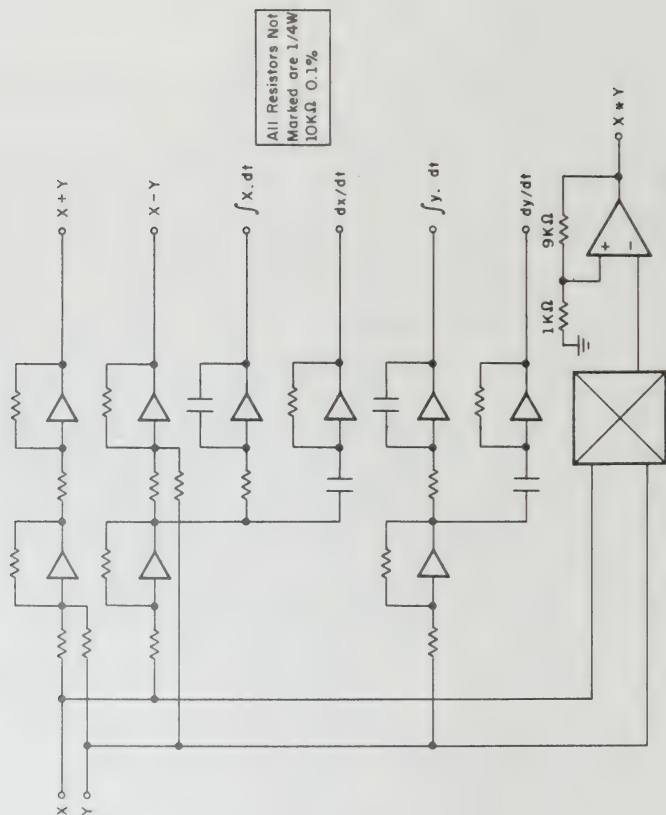


Figure 1. Schematic for a Pure Analog CM

design makes use of straightforward analog circuitry such as operational amplifiers and multipliers. As shown, no division function is included; however, a division scheme can be achieved by connecting three CM's as shown in Figure 2. The integration and differentiation are both with respect to time. Functions of the type

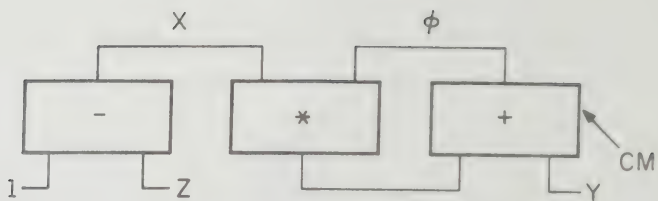
$$\int x \, dy$$

where x is considered to be a function of y , can be achieved using two CM's, as shown in Figure 3.

A quantitative analysis of a pure analog CM was carried out. The op amps were Analog Devices TM type AD 741K and the multiplier a type AD530J. The results are summarized in Table I.

One obvious advantage of this type of implementation is that no A/D conversion is needed at the input to a computer, should the raw input prove to be analog in nature. Hence, quantizing noise is not introduced. However, this advantage is somewhat dimmed by the poor accuracy that can be expected from such an implementation. A substantial increase in accuracy would be accompanied by a violent increase in the price of a CM. It should be born in mind that a typical user might wish to purchase 30-50 CMs for general purpose bench top problem solving.

Two further shortcomings of the analog CM are the inability of the module to execute decision functions, and the lack of memory. The former can be overcome at extra cost by adding two comparators to the schematic, together with strobes to each input on the CM. The outcomes of comparisons ($X > Y$ or $Y < X$) can then be used to enable a certain branch in the program. The latter shortcoming might be overcome by using a charge control memory. However, this is outside the scope of the present project.



$$\theta = X\phi$$

$$\phi = Y + \theta$$

$$\phi = Y + X\phi$$

$$\phi = \frac{Y}{1-X}$$

$$X = 1 - Z$$

$$\phi = Y/Z$$

Figure 2. A Division Scheme

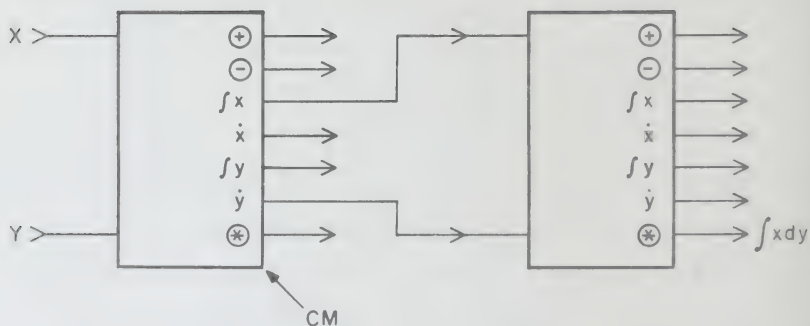


Figure 3. Creating New Functions Using Two CM's

1.6.3 A Stochastic CM

Several important considerations ruled this approach out. As a consequence, evaluation of a stochastic CM is not included in Table I.

The problem of attrition associated with present stochastic arithmetic schemes introduces a factor of $1/2$ at each stage of computation. In a program with any appreciable chain of CMs (>5), this would have the effect of severely restricting the bandwidth [see reference 1]. This problem can be overcome by having an A/S (analog-to-stochastic) converter at the inputs of the CM and an S/A converter at the outputs. This introduces costs far outweighing those saved by the simple nature of stochastic arithmetic. Considerable "conversion" noise is also introduced.

A further problem arises in trying to implement the differentiation function, stochastically. This is very complex and severely reduces the cost effectiveness of the CM.

The CM as proposed is not a system with a high degree of parallelism, which is the type of system that lends itself to stochastic implementation. Hence, the advantages of a stochastic scheme were occluded by the other problems outlined above.

1.6.4 A DDA Type CM

A diagram of a proposed CM is shown in Figure 4. $X(T)$ and $Y(T)$ are 8 bit input signals, which are sampled every ΔT seconds. $\Delta X/\Delta T$ and $\Sigma X \Delta T$ correspond to differentiation and integration respectively. As was seen in section 1.6.2, $\Delta X/\Delta Y$ and $\Sigma X \Delta Y$ type functions can be synthesized using two CM's.

The module may be implemented using 7400 series logic. A

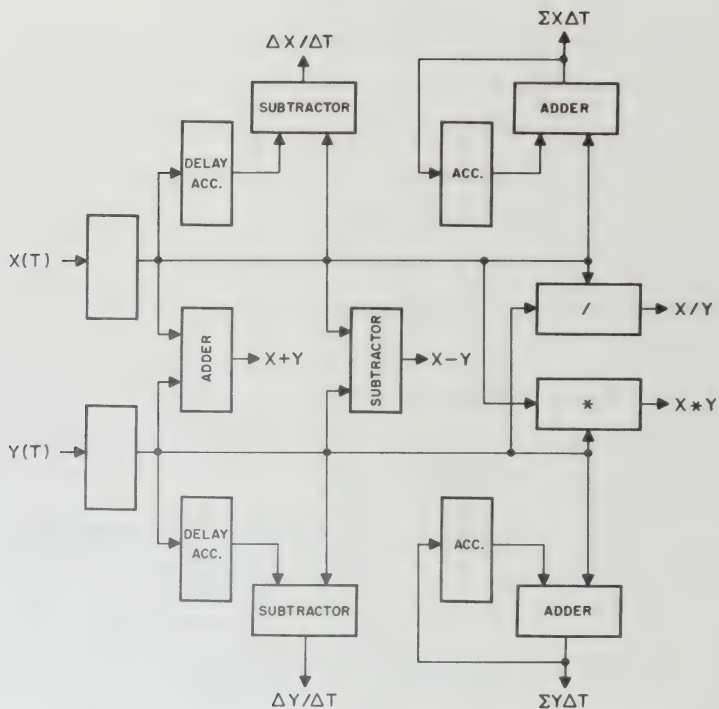


Figure 4. A DDA Type CM

summary of the performance of such an implementation may be found in Table I.

The speed of the CM is limited by the multiplier action. A shift and add scheme is proposed which takes 8 time periods. A clock of 10MHz is used to control the multiply and divide action, and it is divided by 8 to give the sample vote (ΔT). (10MHz is a conservative bound on the speed of 7400 series logic).

Decision functions, together with logical functions can be readily included in the basic CM analysed. Simple memory capacity (such as a stack, or shift register), may also readily be added.

1.6.5 A CM Using a Microprocessor

Two currently available microprocessors were examined for the job. They were the Fairchild PPS25 system, and the Intel MCS-4 system. Because the design of the MCS-4 system was geared more to the handling of I/O instructions, it was chosen over the PPS25. [In order for the CM's to be simply linked together during the programming phase, it is advantageous to have good I/O capability].

The summary of the project performance of an MCS-4 system is shown in Table I. This is a basic memory-less set up using just a CPU and ROM as controller. With the microprocessor approach all the desired functions are not furnished simultaneously. They must be set by some external switch.

1.6.6 Conclusion

The results of the design study pointed to the adoption of the microprocessor type CM. Several of the MCS-4 system are presently being

obtained for bench top testing.

Trevor Mudge

Reference

1. D.C.L. Report #332.

	Cost (\$)*	PWR(W)	Speed(kHz)	Accuracy %	Fan-Out /output
Analog	70	1.1	10	0.8-2	5
DDA	70	16	1MHz	8 bits	10
MCS-4	60	0.8	10**	8 bits	-

Assume op temp range $\pm 10^{\circ}\text{C}$ of room temp.

*No power supplies or A/D converters included in cost estimate.

**Assumes a 10 instruction block of micro code / function.

TABLE I

1.7 Thespiac (Project No. 47)

This quarter has been devoted to two areas of the project - determining the functional characteristics required by the Krannert Center and searching for suitably priced components for the actual construction.

1.7.1 Functional Characteristics

The Krannert Center contains two theatres which are used primarily for dramatic productions. These two theatres are currently equipped with identical lighting control systems which are adequate but cumbersome. Each system contains sixty solid-state dimmers controlled by dc signals (between 0 and 20 volts) from a lighting console. The lighting console provides memory for dimmer-control settings by means of six ports which accept circuit cards containing thirty slide pots. These cards are large (6" x 18" x 1"), difficult to store, and frequently fail to make proper contact when inserted in the lighting console. It is the purpose of this project to replace the current mechanical card memory with a digital memory and to increase the flexibility of the control console.

The current console will accept six cards (30 settings each) at any time, thereby providing three complete scene settings (60 settings each), or cues, on-line. The console allows each dimmer to be individually disabled, assigned to the card memory, or placed under manual control (a permanent bank of 60 pots in the console are used for manual control). In turn, the card memory may be placed in one of two modes: the crossfade mode or the submaster mode. In the crossfade mode, the console allows the operator to gradually lower all settings in one cue

while raising all settings for a second cue. In the submaster mode, each of the three cues is controlled by a submaster which can raise or lower all of the settings for its respective cue. Krannert Center has found the mutually-exclusive nature of the crossfader and submasters and the permanent assignment of submasters to entire cues to be overly restrictive.

As it currently stands, the digital-memory system will provide a crossfader, five submasters, and storage for 256 cues. Each dimmer may be assigned to any of the submasters or the crossfader; on a given dimmer, the crossfader and submasters will still be mutually exclusive.

Figure 1 shows a block diagram of the control required for one dimmer. This model provides independent (by dimmer) submaster selection, automatic submaster selection under memory control, an easy method to allow settings to be made digitally (rather than providing an analog pot with an A-to-D converter), and lends itself to parallel control. Figure 2 shows the system overview.

The "control unit" block in Figure 2 will contain the five analog submasters, the analog crossfader, and the digital logic necessary to govern the sixty dimmer controls. The digital portion of the control unit will primarily control interactions between the "main memory" and the individual memory buffers in the dimmer controls. Two levels of memory protection will be provided: the memory as a whole may be made read-only, or active locations may be made read-mostly. The read-only mode is necessary to prevent unauthorized modification of the settings and to safeguard the memory during live performances. The read-mostly mode is provided for use during show set-up; it warns the operator if he attempts to write into a main memory location that contains a cue. If

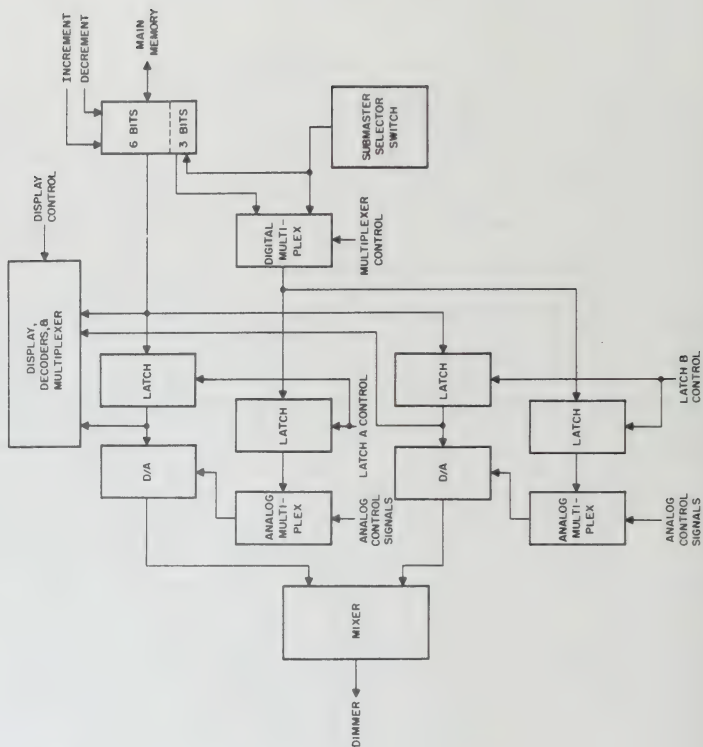


Figure 1. Block Diagram of a Dimmer Controller

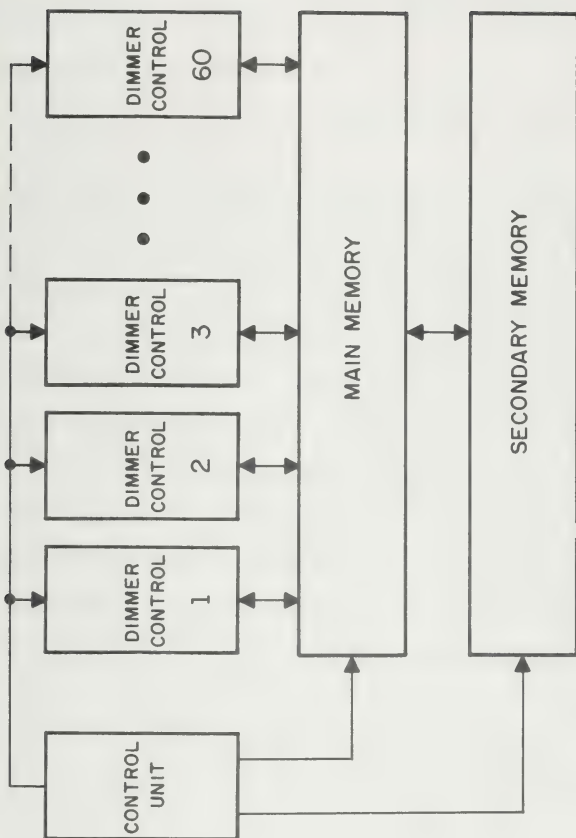


Figure 2. Thespiac Overview

he is attempting to modify that cue, he may override the warning and store the updated setting. If he does not wish to modify the existing cue, he may choose another memory location and store there. The control unit will also provide for automatic and manual cue sequencing.

1.7.2 Component Search

A wide variety of components will be needed for the construction of Thespiac. In order to make cost estimates, the components which seemed to be most expensive or most difficult to locate have been investigated.

Four major choices were available for the main memory: magnetic cores, plated-wire, discs, and semiconductor memories. The storage capacity required for Thespiac is 147456 bits. Of the aforementioned memories, the fixed discs were least expensive on a per bit basis (approximately \$2100 for 800K bits). This form of memory has been rejected for reliability reasons. (The intended operating environment is not dust free and the disc systems are not sealed). Plated-wire and semiconductor memories were both expensive (relative to cores and discs) and are not being considered for the main memory. Core storage in an 8K x 18 organization is priced around \$2500 from several vendors (including chassis and power supply) and is the current choice for the main memory.

The secondary memory choices considered are cassette tape drives and 'floppy' discs. Prices of these two types of devices are competitive at around \$700.

In the dimmer control circuits, major cost contributors appear to be the D-to-A converters, the analog multiplexors, and the display.

A suitable D-to-A converter is made by Motorola for \$4 in quantities over 100; no other D-to-A converters in this price range have been located. The analog multiplexors are still being researched, but it is expected that their price will be less than \$4 each. The display requirements are not yet settled and cost estimates have not been made. At this stage, however, it appears that two digits per dimmer will be required.

1.7.3 Summary

Work has progressed to the point where a formal agreement on the design goals can be written during the early portion of next quarter. Upon the acceptance of these goals by Krannert Center, details of the design will be finalized, complete circuit drawings made, and construction will then begin.

L. N. Daley

2. HARDWARE SYSTEMS RESEARCH

Summary

1. Lascot (Cooper)

System improvements concerning beam deflection, screen quality and laser.

2. LINDA (Blandford)

Circuit design for video amplifier, double differentiator and delay line for shading feature.

3. Stereomatrix (Whiteside)

Increase in precision of coefficient generator, modification of laser.

4. Scantrix (Yuen)

Power switch for LED display.

5. Frog (Bose)

Progress in simulation.

6. Beacon (Jer)

Synchronization of sequential freeze circuit.

7. Caecotron (Tse)

A new project aiming to develop a scanning device to produce an audible representation of a visual scene to be used by the blind.

8. CM - Correlation Modulation (Kowall)

Also new, a method of transmitting information by the time delay between two carriers that need not be sine waves. Prescribed varying of the delay allows secret communications.

M. Faiman, Editor

2.1 Lascot (Project No. 09)

A vertical deflection mirror (General Scanning G0606) was ordered and received. It has a vibrating bandwidth of 3 kHz and takes a drive current of 300 mA/degree. The vertical deflection driver circuitry was redesigned and the mirror was installed. A high gain (≈ 3) screen was also installed to improve the brightness of the picture within the viewing angle of 20° .

A wooden enclosure was prepared for the Lascot optics to prevent (1) stray light escaping in the otherwise dark room, and (2) dust particles settling on the exposed optical surfaces.

The laser showed reduction in its power output, attributed to the soft coated mirror. A hard coated mirror, developed by Spectra Physics, was ordered.

M. N. Cooper

2.2 LINDA (Project No. 28)

2.2.1 Summary

The ultimate goal of project LINDA (LINE Drawing Analyzer) is to build a machine capable of recognizing a small vocabulary of line drawings independent of their size and orientation (within reasonable limits). Line drawings will consist of polygons and circles made up to form familiar figures. Recognition is based on the premise that a fairly accurate guess as to a drawing's identity may be made if one can determine what the simple parts of the drawing are and their relative location with respect to one another. For further details of operation refer to the quarterly report for Jan., Feb. and March, 1972.

2.2.2 Project Status

2.2.2.1 Video Amplifier

The video amplifier shown in Figure 1 was completed and tested during the past quarter. It takes a TTL input and provides a 0 to 45 volt output capable of driving the z-axis scope terminal at frequencies up to about 6 MHz. This operation is more than sufficient for the display in this system.

2.2.2.2 Double Differentiation

It will be recalled that polygon identification is performed by summing the ordinates of a displayed polygon to form a sum signal having discontinuities corresponding to those in the polygon. The sum signal is easily differentiated twice to obtain pulses from the discontinuities which are unique for a given polygon. Figure 2 shows the double differentiation system for LINDA. The differentiator has an upper cut-off frequency of about 70 kHz to help cut down on noise inherent in such systems. The double differentiator is followed by a circuit which takes the absolute value of the second derivative to give positive pulses corresponding to each discontinuity in the original sum signal.

2.2.2.3 Delay Line

Other work this quarter included the building of an improved delay line. A delay line had been in use which used one-shots to delay up to 8 pulses on a video line for 63.5 μ sec. Preliminary work on a system to shade in or erase selected parts of a figure indicated that more than one line delay would be needed. The one-shot delay line

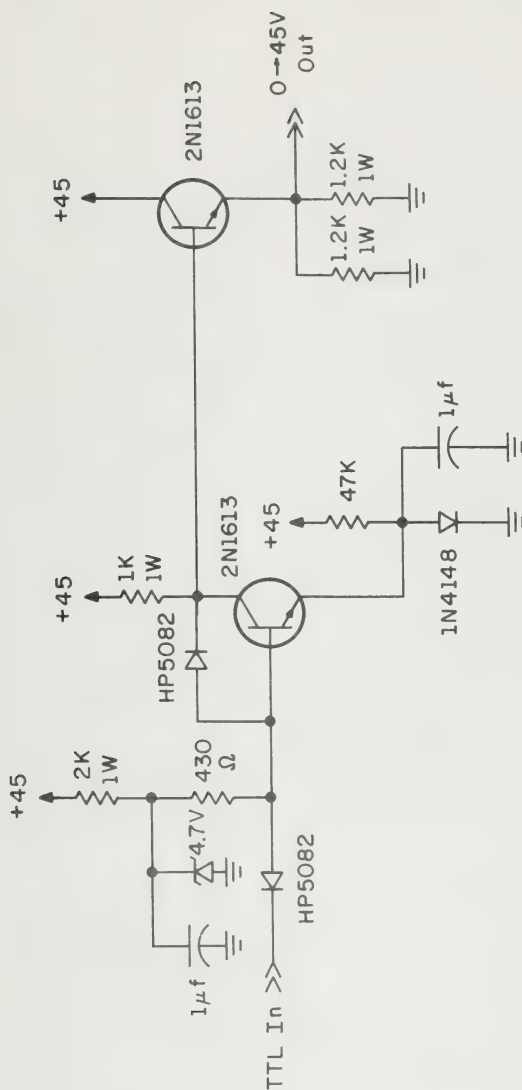


Figure 1. Video Amplifier

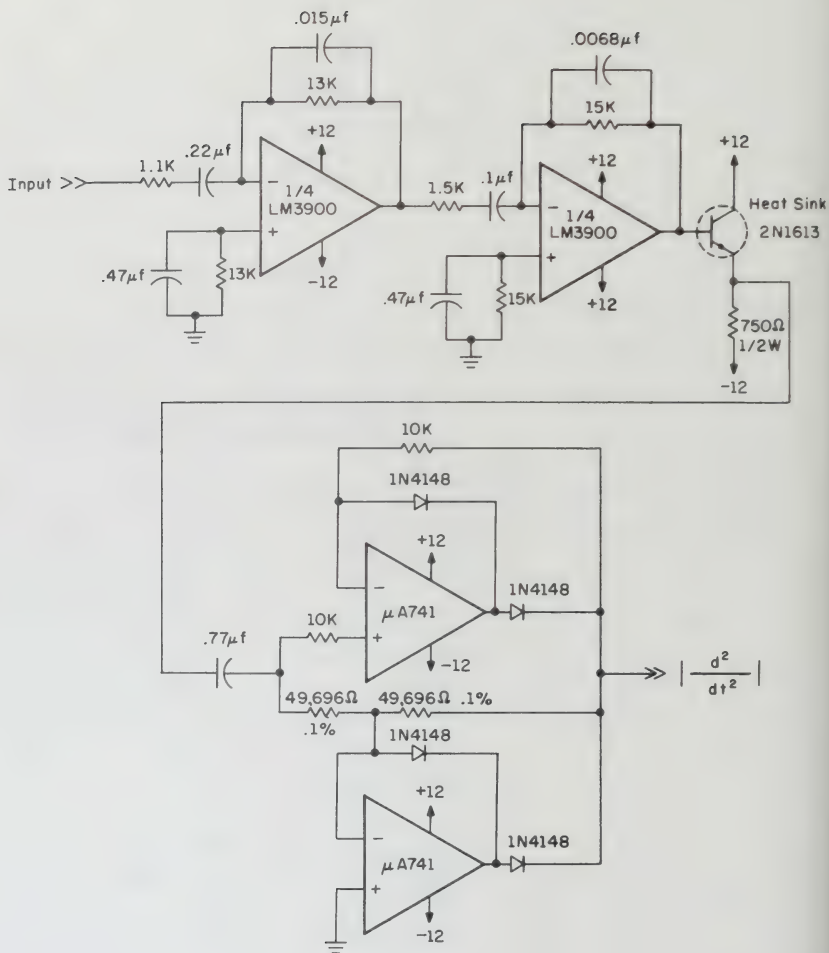


Figure 2. Circuit for Taking 2nd Derivative and Absolute Value

arrangement demanded much adjustment and several delay lines of this nature would probably become impossible over a period of time. A shift register delay line has since been built. It is shown in Figure 3. Three companies were found which build 512 bit shift registers capable of the 8 MHz data rate needed. The Intel 2503V was chosen for its simplicity and low cost. The 2503V features dual 512 bit registers which can take data at an 8 MHz rate on a 2 phase clock. This type of delay line is easily serialized to get a delay of whatever length is desired.

2.2.2.4 Shade/Erase Logic

At the present time a system has been built which uses the double delay line of Figure 3 to shade in or erase any part of a line drawing made up of multiple polygons. Basically, the system requires an initializing pulse to indicate which polygon is to be shaded in and which to be erased. Using the delay lines, successive video lines are compared and shading continues down the figure along the polygon initialized. Although all the cards have been made up the system has not been debugged or tested and results and further details will be discussed in future reports.

2.2.3 Future Work

In the next quarter work will continue on the shade/erase logic until a suitable system is obtained. LINDA will then be made automatic so that she can sequence through a given drawing, shading in successive polygons for identification, without manual operation.

Dick Blandford

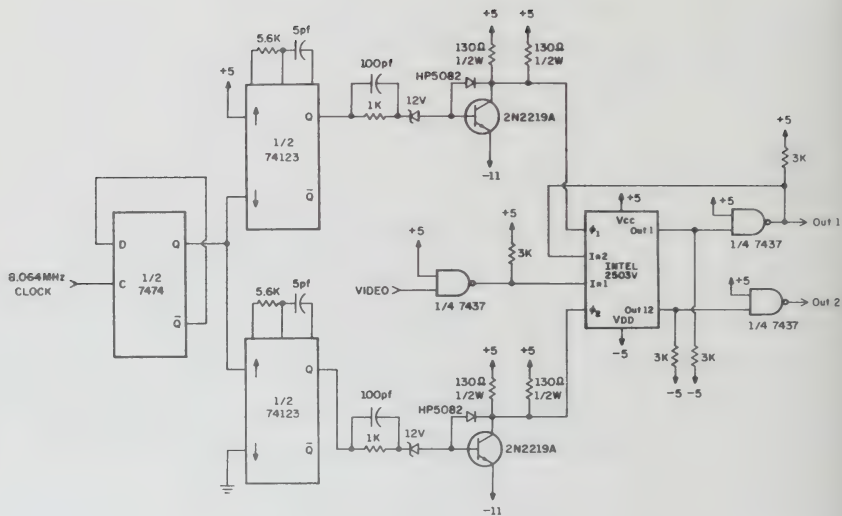


Figure 3. Shift Register Delay Line with 2 Phase Clock

2.3 Stereomatrix (Project No. 30)

2.3.1 Coefficient Generator

During this quarter the extension from nine to twelve bits plus sign was completed. Now two-degree increments of rotation are satisfactorily performed. One-degree increments were hoped for but cause a very noticeable error of about 10% after one revolution. For these small angles, the accuracy remaining after N iterations seems to be $\frac{2^{12}}{N}$.

Occasional noise from the control box forward and backward switches has been eliminated by inserting clocked flip-flops prior to the decoder on the control card which are clocked by the frame pulse at about 20-400 Hz. The flip-flop ignore any noise butsts but cause no apparent delay in control response.

A lock-up problem was discovered on the control card which occurred occasionally at turn-on and when switching to SLOW operation. The slow divider would allow improper retriggering of the Frame Inhibit flip-flop. This was corrected by simply triggering the Frame Inhibit flip-flop from the Delayed Frame pulse only.

Scaling circuit problems resulting in jumpy operation and quivering output have been traced to noisy input pulses. These must travel over 3 feet from the control card. Rerouting the Delayed Frame pulse to arrive via a flip-flop and thereby synchronizing it with the control card clock eliminated the erratic jump problem. Simple R-C filters on the transmission lines may solve the quiver problem.

The inverse transformer was assembled, multipliers adjusted, and it is now partially working. Voltage levels are not yet what they should be. This will be corrected soon.

2.3.2 Display

The laser now seems to run satisfactorily with the added heat sink containing 22 transistors. The new start circuit seems to work very well. A two-diode, two-resistor easy-on circuit will be added soon to reduce the high current surge at turn-on.

Steve Whiteside

2.4 Scantrix (Project No. 35)

Referring to the LED display unit and its driving circuit described in the last quarterly report, it is obvious that a switching scheme is required to switch the horizontal lines of LEDs on and off, one row at a time. An easy and inexpensive way to do this is to build 64 simple power supplies. Figure 1 shows a 5V, 1.5A power supply. It is estimated that each of these supplies will cost less than \$10.

The first section of the LED panel with the driving circuits is almost finished. As soon as this section is tested, the rest of the panel can be built. It is expected that this project will be completed before June.

Sik Yuen

2.5 Frog (Project No. 36)

At the beginning of the quarter, the simulator for "Frog" was used for a final simulation. The latter weeks were devoted almost completely to the analysis of the results of the simulation and the documentation of the entire study. We hope to wrap up these simulation studies shortly with the publication of a detailed description of the work done thus far in the form of a DCS report. Therefore, no further

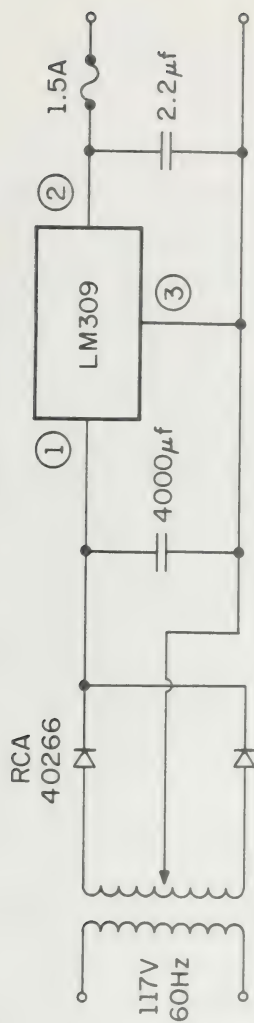


Figure 1. 5V, 1.5A Power Supply

elaboration is attempted here.

For the hardware implementation a logical design of the T* element has been made. Although parts of this design were experimentally verified, the circuit is yet to be finalized.

Dev Bose

2.6 Beacon (Project No. 38)

The goal of project Beacon is to display a TV picture on a 32 x 32 LED square matrix. A square portion of a TV picture is used and is divided into 32 x 32 small areas. Each small area is monitored by a photo transistor. The signal from the photo transistor is amplified to drive a LED.

A 17 inch black and white TV monitor was chosen for the project. The vertical and horizontal sync signals are taken from the TV and fed to a buffer amplifier whose output is used to drive a Schmitt trigger (Figure 1). These two signals are used as inputs to the sequential freeze circuit described in the last quarterly report.

The other change to the TV set is that the wire connected to the wiper of the brightness control was cut and two wires brought out. These are used to turn the electron beam of the picture tube on or off.

Martin Jer

2.7 Caecotron (Project No. 43)

During the last quarter, experiments were done to determine the ability of the ear to distinguish tones and tone sequences. It was found that for a given percentage change in the frequency, the ear can more readily detect a change for a square wave than for a sine wave.

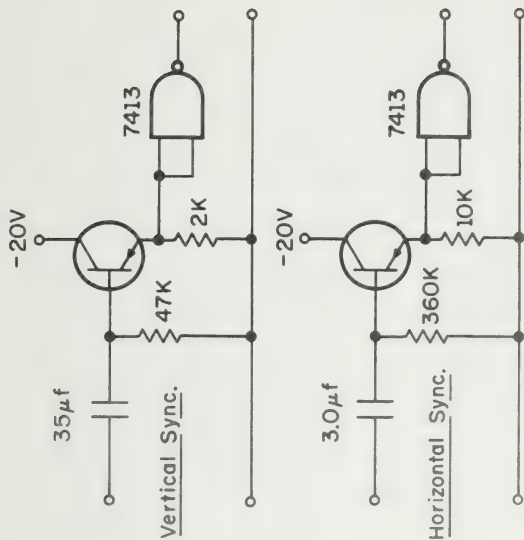


Figure 1. Sync for Sequential Freeze

This is not unreasonable since a change in the repetition rate of a square wave creates a change in each of its overtones. For this reason, square wave tone sequences were chosen for the audio signal of the system.

It was also decided that the audio signal should fall within a fairly narrow range of the audio spectrum so as to accommodate people with limited hearing abilities. An upper frequency of about 2112 Hz, a C, was chosen for the 1st line of the audio picture. 15 other tones, each one lower in frequency than the one preceding and within a few percent of a corresponding note in the equally tempered scale, were chosen as the fundamental frequencies of the other 15 lines. In this case, the lowest note would be a B at 495 Hz. Such a tone sequence would easily fall within the hearing range of most people; and if these tones are played in sequence, they should be easily recognized by non-musical operators. A table of these tones and their frequencies is shown in Figure 1.

As described in the previous report, a 'white' object on a 'grey' background would produce a slightly lower tone to that of the fundamental frequency of the line the object is situated in. Correspondingly, a 'black' object would produce a slightly higher tone. It was discovered that a change of $1/64$ of the fundamental frequency (less than a half tone) can be noticed without much difficulty. This is true only for fundamental frequencies lower than about 6 kHz.

A tone generator that meets the above specifications was designed and built during the past quarter. A diagram of the circuit and its controls is shown in Figure 2. It has a pulse train generator with a repetition rate of about 10 MHz. The frequency of this pulse train can be modified by a series of binary rate multipliers, each capable of

Tone	Frequency f	Multiplying Rate	Control Signals						
			X	$\frac{X}{2}$	$\frac{X}{4}$	C	B	A	G F E
C"	2112	C" x 1	1	0	0	1	0	0	0 0 0
B'	1980	C" x $\frac{60}{64}$	1	0	0	0	1	0	0 0 0
A'	1760	B' x $\frac{57}{64}$	1	0	0	0	0	1	0 0 0
G'	1584	A' x $\frac{58}{64}$	1	0	0	0	0	0	1 0 0
F'	1408	G' x $\frac{57}{64}$	1	0	0	0	0	0	0 1 0
E'	1320	F' x $\frac{60}{64}$	1	0	0	0	0	0	0 0 1
D'	1188	E' x $\frac{58}{64}$	1	0	0	0	0	0	0 0 0
C'	1056	$\frac{1}{2}$ x C' x 1	0	1	0	1	0	0	0 0 0
B	990	$\frac{1}{2}$ x C' x $\frac{60}{64}$	0	1	0	0	1	0	0 0 0
A	880	$\frac{1}{2}$ x B' x $\frac{57}{64}$	0	1	0	0	0	1	0 0 0
G	792	$\frac{1}{2}$ x A' x $\frac{58}{64}$	0	1	0	0	0	0	1 0 0
F	704	$\frac{1}{2}$ x G' x $\frac{57}{64}$	0	1	0	0	0	0	0 1 0
E	660	$\frac{1}{2}$ x F' x $\frac{60}{64}$	0	1	0	0	0	0	0 0 1
D	594	$\frac{1}{2}$ x E' x $\frac{58}{64}$	0	1	0	0	0	0	0 0 0
C	528	$\frac{1}{4}$ x C' x 1	0	0	1	1	0	0	0 0 0
B	495	$\frac{1}{4}$ x C' x $\frac{60}{64}$	0	0	1	0	0	0	0 0 0

Figure 1. Cascotron Tone Frequencies

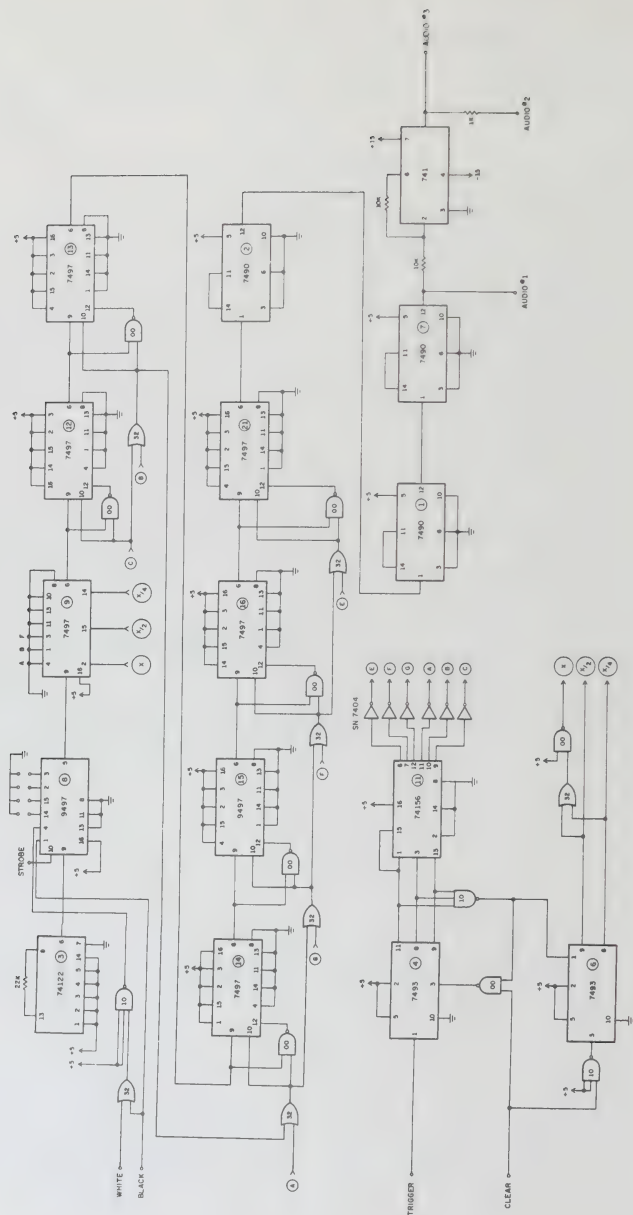


Figure 2. Tone Generator

reducing the incoming pulse train by a factor of $\frac{x}{64}$ (i.e. $f_{out} = f_{in} \times \frac{x}{64}$). By using the appropriate control signals, as shown in Figure 1, a tone with average repetition rate of $f \times 10^3$ can be obtained at the output of the last multiplier. To get a pulse train with a uniform repetition rate and duty cycle, this signal is divided by 10^3 and amplified to drive a small speaker or a headset.

Using this tone generator and a simulator that can produce the audio picture of a door, a tall fence, a waist-high fence, etc., an investigation was conducted with the help of volunteers to determine the maximum rate at which the audio picture could be displayed. It was found that a rate of 1 line per second (which corresponds to 16 seconds per audio picture of 16 lines) could still be deciphered. The line rate of this tone generator is adjustable by the operator, who can select the display rate that he prefers.

During the next quarter, work will be done on rangefinding and object identification to produce the controls to drive this tone generator.

B. Tse

2.8 CM (Correlation Modulation, Project No. 48)

2.8.1 Introduction

This is the first report on a new project in the Hardware Research Group dealing with a new form of information transmission. Information is usually transmitted by varying some characteristic of a carrier signal (e.g. a high frequency sine wave) in a way which is dependent upon the information to be sent. In amplitude modulation (AM), in frequency modulation (FM), or in phase modulation (PM), the amplitude, frequency, or phase, respectively, is varied in a way which is linearly

related to the changes in the signal (information) which we wish to transmit.

Correlation Modulation can be considered a form of phase modulation, but it goes one step beyond the traditional modulation techniques by encoding the information to be sent in the time delay between two signals. The two signals can be, for example, a sine wave carrier and a delayed version of this carrier signal.

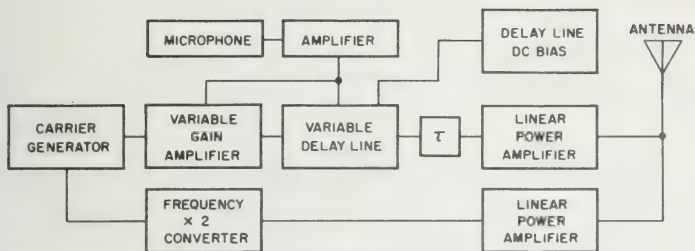
The idea of encoding information in the time delay between two signals is not entirely new. In World War II, the Wilcox communication scheme used two standard time delays to transmit the "dots" and "dashes" of Morse Code. Correlation Modulation goes beyond this by employing a variable delay line, with a continuous range of delays, to encode information by continuously changing the delay of the carrier signal, just as information is encoded by continuously changing the carrier amplitude in amplitude modulation. Secret communications are an obvious application for this type of modulation.

2.8.2 The Correlation Modulation System

Phase I of the Correlation Modulation Project will be to design and build a Correlation Modulation transmitter-receiver system to take audio input from a microphone and encode this signal in the time delay between a sine wave carrier and a delayed version of this carrier. A system block diagram is shown in Figure 1.

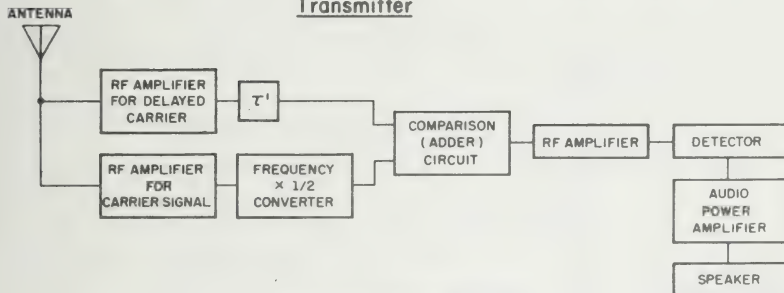
2.8.2.1 The Delay Line

The heart of this system is the variable delay line. It consists of ten LC delay sections. The capacitors are formed from the



$$T = (\text{DELAY OF FREQUENCY CONVERTER}) - (\text{DELAY OF VARIABLE AMPLIFIER})$$

Transmitter



$$T' = \text{DELAY OF FREQUENCY CONVERTER}$$

Receiver

Figure 1. Correlation Modulation Phase I System Block Diagram

depletion region capacitance of back-biased diodes. The capacitance of each of these diodes varies continuously with the applied back-biasing voltage. Thus a continuous sequence of delays is available. A circuit diagram of this delay line is shown in Figure 2.

2.8.2.2 The Microphone Amplifier and Delay Line DC Bias

An operational amplifier microphone amplification circuit for the high output impedance microphone used is shown in Figure 3. Also shown in this figure is the way a DC bias is applied to the delay line.

2.8.3 The Correlation Modulation Test Experiment

The major work done this quarter has been the building and testing of a mock-up Correlation Modulation system. The purpose of this system has been twofold: 1) to see if variable delay information transmission is feasible and 2) to design and build some of the associated Correlation Modulation circuitry. A block diagram of the test system is shown in Figure 4. The microphone amplifier and delay line circuitry are those shown in Figures 2 and 3. A standard radio receiver is used as the final block. The system operates at a frequency of 1 MHz. The comparator and its associated amplifier are rather elementary and their circuit diagrams have not been included in this report.

This system does work and thus it verifies the feasibility of Correlation Modulation. There are, of course, some system problems. One of these is distortion due to the nonlinear attenuation characteristic of the delay line. The effect of this distortion will, hopefully, be greatly reduced by the variable gain amplifier which precedes the delay line in Figure 1. The solution to this problem and the design of circuitry for the unconstructed blocks in Figure 1 will be the subject of Correlation Modulation research in the next quarter.

Richard J. Kowall

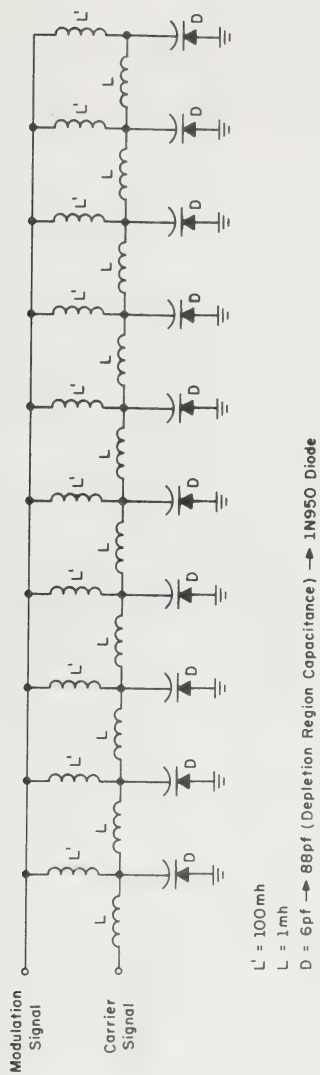


Figure 2. Variable Delay Line

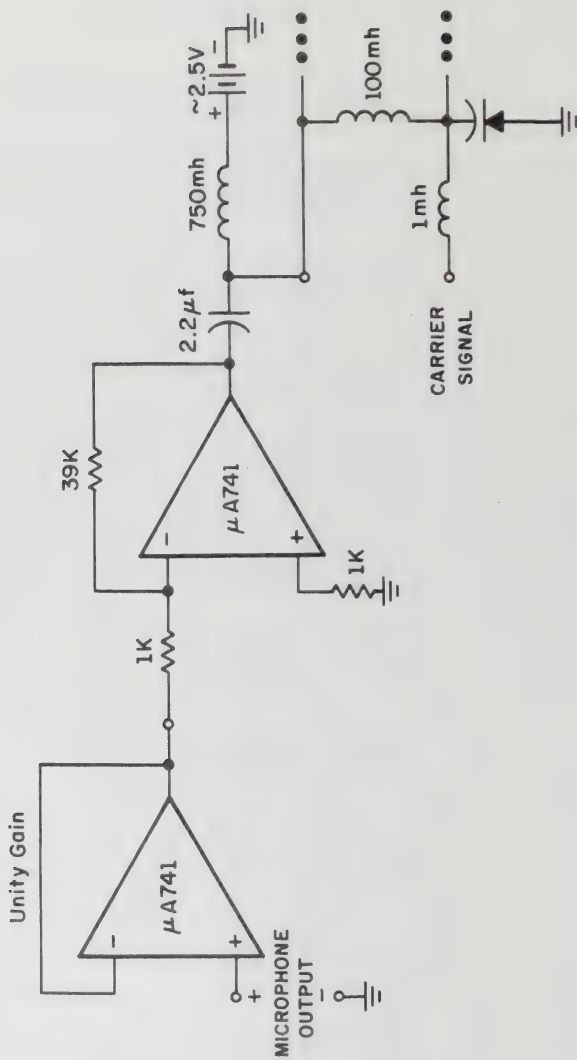


Figure 3. Microphone Amplifier and Delay Line DC Bias

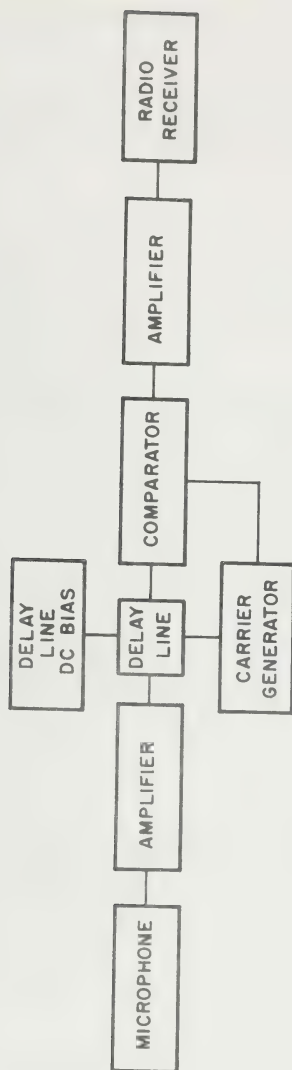


Figure 4. Block Diagram of the Correlation Modulation Test System

PUBLICATIONS

Shiv Prakash Verma, "Perspective Transformer for the Stereomatrix 3-D Display System," Department of Computer Science Report No. 532, University of Illinois, Urbana-Champaign, October 1972.

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO. C00-1469-0218	2. TITLE 4th Quarterly Progress Report 1972
--	--

3. TYPE OF DOCUMENT (Check one):

☒ a. Scientific and technical report

☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

☒ a. AEC's normal announcement and distribution procedures may be followed.

☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.

☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

SUBMITTED BY: NAME AND POSITION (Please print or type)

W. J. Poppelbaum
Professor and Principal Investigator

M. Faiman
Professor of Computer Science

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature

W. J. Poppelbaum

Date

December 31, 1972

FOR AEC USE ONLY

AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION
RECOMMENDATION:

PATENT CLEARANCE:

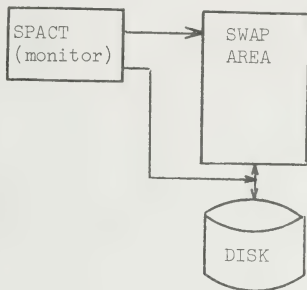
☐ a. AEC patent clearance has been granted by responsible AEC patent group.

☐ b. Report has been sent to responsible AEC patent group for clearance.

☐ c. Patent clearance not required.

3. SOFTWARE SYSTEMS RESEARCH

In order to develop a system capable of supporting more than one user, swapping is being implemented in the existing simulation system. The general system structure is diagrammed as follows:



The monitor controls execution in the swap area and controls swapping to the disk. Work on the monitor and the FETCH routine used to load programs initially from a standard OS/360 program library and on the routine COMMUNE which builds picture and text blocks is described in this report.

Also discussed is work on the interpreting system for the PDP-8 to permit programming its graphic functions in a higher level language. Other reports include work on improving the efficiency of DIFSUB, the numerical integrator, and ILLISM, the graphic simulation system for the PDP-11.

3.1 Numerical Processes

3.1.1 DIFSUB (R. L. Brown, M. Ostrar)

A method was investigated for evaluating the efficiency of proposed new versions of DIFSUB and other multivalued methods relative to the present version of DIFSUB. A collection of test problems was chosen to provide benchmark results on which to base comparisons. The current version of DIFSUB, soon to be available in the UOILIB library as DIFSBZ, was run on each test and the results were tabulated. DIFRK, a program incorporating a fourth order Runge-Kutta starting and restarting procedure, is being run to compare its performance with DIFSBZ.

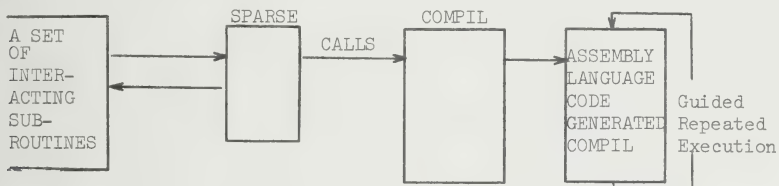
A rational arithmetic package, designed to carry out addition, subtraction, multiplication, and division on rational numbers, was written. Using this package and a recurrence relation due to L. F. Shampine and M. K. Gordon, the coefficients $A(I)$ for the Nordsieck form of the Adams method were computed as rational numbers. This will allow the higher order Adams method in DIFSUB to have coefficients computed to the maximum accuracy of the machine it is used on, thus increasing program portability.

A program written at Oxford University is designed to compile input systems of ordinary differential equations with initial values into output FORTRAN programs to evaluate the system using Taylor's method. A set of problems--some used to test versions of DIFSUB, and some others--was compiled and the FORTRAN programs were sent here for comparison of CPU time used in execution. Using the UOILIB library routine, STIMEZ, to evaluate the CPU execution time after all initial values were set up, gave good approximations to run time for different

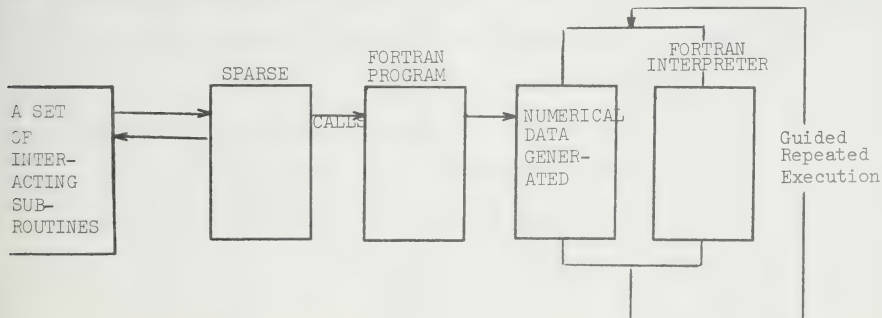
error per step criteria EPS. In general, the Taylor's method took less execution time and the final result was more accurate than for DIFSUB.

3.1.2 SPARSE (J. Deogun)

Many variants to the present numerical package have been proposed. The present system could be roughly described in three parts for our purposes. 1) A set of interacting subroutines--one of which guides the execution of SPARSE, 2) SPARSE, 3) COMPIL. In the event of execution, SPARSE calls COMPIL and COMPIL generates assembly language code which forms the subroutines MATIN1, MATIN2, MATIN3, and MATMUL. All of these subroutines are executed repeatedly to improve on the matrix inverse.



COMPIL is a fairly complex program, and code generated by COMPIL is usually very long. Another way to do what COMPIL does is to generate coded numerical data by a simple FORTRAN program which replaces COMPIL. The numerical data so generated is interpreted by a simple FORTRAN interpreter. The FORTRAN program and the interpreter described above are simple as compared to COMPIL, and the numerical data generated is short.



Subroutine SPARSE uses the Gaussian elimination to solve the system $Ax = b$. A second variant is, therefore, to use CROUT decomposition instead of Gaussian elimination. There are many more variants, but these are the two being investigated this quarter.

3.1.3 Remote Data Structure Utilities: COMMUNE (B. van Melle)

COMMUNE is in the process of being revised in anticipation of the new system. The changes are due largely to limitations in the amount of available core. The facilities will be more extensive than those of the original COMMUNE (as described in DCS Report #466), but have been simplified from those of the generalized package (QPR's beginning 4th quarter 1971).

The new routines will perform GETMAIN's only for the large blocks of core needed to store picture blocks. The initialization routine CALLER will procure one block of core to be used for buffers, etc. The user will only be able to manipulate one line block and one text block at a time, but he will have ready access to the filing system for storing and retrieving blocks. The routines presently planned for the new COMMUNE are as follows:

```
CALL MESSAGE(TEXT,LEN)
CALL SEND(IBLK,ICODE)
```

These routines send messages and initialized blocks, respectively, to the terminal, and they operate the same as originally.

```
CALL REPLY(ITYPE,LEN,IDATA[,NCHARS[,LINNUM]])*
```

Gets input from the terminal. The parameters are as before, with the addition of LINNUM, which when present returns the number of the last line on the screen. Also, if the input was a joystick hit

* Brackets [] enclose optional parameters.

(ITYPE=1), IDATA(4) contains the pushbutton number (IDATA(1-3) contain, as before, the X and Y coordinates of the hit and the screen segment).

CALL MASK(IMASK)

Filters terminal input. The only change is the addition of a twelfth element to IMASK:

IMASK(12) = 0: Modify X,Y coordinates according to pushbuttons.

1: Do not allow this modification.

CALL ERASE(IFRAME[,IREGEN])

Erases the screen. If IFRAME \neq 0, a frame is put up. If IREGEN is present and IREGEN \neq 0, the current data structure is redrawn.

CALL LINE(X,Y,INTEN[,&STMT])

If INTEN > 0, a line from the previous point to (X,Y) is entered in the line block. If INTEN = 0, no line is drawn, but (X,Y) becomes the new point. If INTEN < 0, a new entry is created in the block, and (X,Y) are the increments applied to all coordinates in the entry (ordinarily the increments are 0). If the line block fills up on this call, or was already full, and the line block dump option is not in effect, control returns to &STMT via a RETURN 1.

CALL LINED(DX,DY,INTEN[,&STMT])

Same as LINE, except (DX,DY) are increments added to the old endpoint to determine the new.

CALL TEXT(X,Y,TEXT,LEN[,&STMT])

Enters text line in text block, (X,Y) coordinates of the first character. If X = Y = 0, the new line appears immediately below the previous one if LEN > 0, or is concatenated to the previous one if LEN < 0. Consecutive lines may be entered in one call by separating lines with '@' (denoting carriage-return, line feed).

Each line appears below the previous one. '@@' would cause a blank line to appear. If the text block is full or fills up while trying to enter this text, control returns to &STMT.

CALL CLEAN(IBLK)

Cleans out the indicated block:

1 = line block
2 = text block
-1 = both

The next four routines utilize the filing system and have the following common parameters:

IBLK - chooses one or both blocks, as in CLEAN above.

NAME - a character string of the form

'<filenames>' or '<filename>/<libraryname>'

or '<filename>/<username>.<libraryname>'.

<filename> may not exceed six characters.

When not specified, <username> is taken to

be the logon name and <libraryname> is PICLIB.

NAMLEN - the number of characters in NAME.

CALL SAVE(IBLK,NAME,NAMLEN,IKEEP)

Saves the specified block(s). If IKEEP = 0, the block(s) are cleaned out after saving them.

CALL FETCH(IBLK,NAME,NAMLEN[,&STMT])

Fetches the indicated block(s). If the block(s) already exist, they are overwritten. If nothing is found, control returns to &STMT via a RETURN 1.

CALL DISPLY(IBLK,NAME,NAMLEN,IFRAME[,&STMT])

Fetches the indicated block(s) and sends them to the terminal. The block(s) are not available for manipulation and any

line or text block(s) currently under construction remain undisturbed.

IFRAME has the following meaning:

IFRAME = 1 erase the screen and put up frame first.

= 0 just erase the screen before displaying

= -1 do neither

CALL DELETE(IBLK,NAME,NAMLEN[,&STMT])

Deletes the indicated block(s). IBLK may range from 1-10, and if IBLK = -1, the entire picture is deleted.

3.1.4 Plot Package (W. Chung)

1. The revision of the plot package has been under continuous progress while the present version has been used in test runs and demonstrations. In order to simplify the program structure and thereby to reduce the response time, all the menu pictures will be stored on XFILE and also user-specified graphical outputs will be stored and fetched to and from XFILE.

To shorten the programming time and remove as many programming errors as possible, a simple picture construction subroutine is written in FORTRAN. This subroutine will facilitate the interactive construction of pictures, particularly whose lines and text lines should be positioned at specific coordinates, e.g., menu and frame. A user at the terminal can pick up one of the commands with the joystick or type in coordinates and/or text lines to be constructed. The calling sequence is

CALL PICON(NPIC,ITEXT,ICODE,IBLK)

2. A few experiments with interpolating methods have been done to produce better looking graphical output. Two sides of the problem are considered to be 1) what interpolation scheme is to be used, and 2) what strategy is to be employed to choose a proper set of interpolating points such that storage requirements and the computation time are minimized.

Three different methods of interpolation are tested:

- a. ordinary cubic spline,
- b. geometric interpolation which uses a third-degree polynomial with geometrically approximated derivative values at data points, and
- c. spline under tension.

When the original data from the test examples NET⁴ and NET⁷B are tried, the CalComp plotter outputs are almost undistinguishable. However, when a data point selection scheme is combined with interpolation, the results are very different. The sensitivity of methods to data-point set in terms of deviation from the original curves can be observed. Experiments with a data-point selection scheme using the derivatives of the original function are underway, hoping to draw some conclusions. The calling sequences for the geometric and spline under tension methods are, respectively:

```
CALL CRVFIT(IU,MODE,ND,X,Y,DY,M,N,U,V)
CALL SPFARM(N,X,Y,XLP1,SLPN,YF,TEMP,SIGMA)
CALL SPUT(T,N,X,Y,YF,SIGMA,IT)
```

3.1.5 Item Analysis (J. Koch)

The instructions for an interpretive language that will run on the PDP-8 were created this quarter. This language will be used to create systems (such as the existing GSAM--generalized simulation and modeling) that will allow a user to use the existing picture data structures in different and easier ways. For instance, many of the functions done now must be explicitly joysticked by a user, but a user could be "lead down the garden path" by making many of these functions implicit and prompting the user. The interpretive language makes it easier to code these systems on the PDP-8 by eliminating the need to code in assembly language and the need to know system conventions and routines. The instructions are simple words or abbreviations that indicate the operations to be performed. Thus, a clearer idea can be obtained of what a program is actually doing than could be gotten by examining the assembly language equivalent.

The bit assignments for the operation (op) codes was done to facilitate speed of decoding on the PDP-8. Thus, an instruction can be placed in the accumulator, a left shift of one done and a test of the link bit to see which branch of the tree in Figure 1 to follow. By examining the first few bits of the op code, the length and type of an instruction can be easily determined.

The language was written so that all variables are forced to be kept in a console vector area. The programs will be assembled and checked for errors by a program in the 360 (see 3.1.6), and that program will also divide the interpreter language programs into 1024 word pages. Because all variables are in the console vector, these pages are not modified and can be overlayed without having to be swapped out to disk.

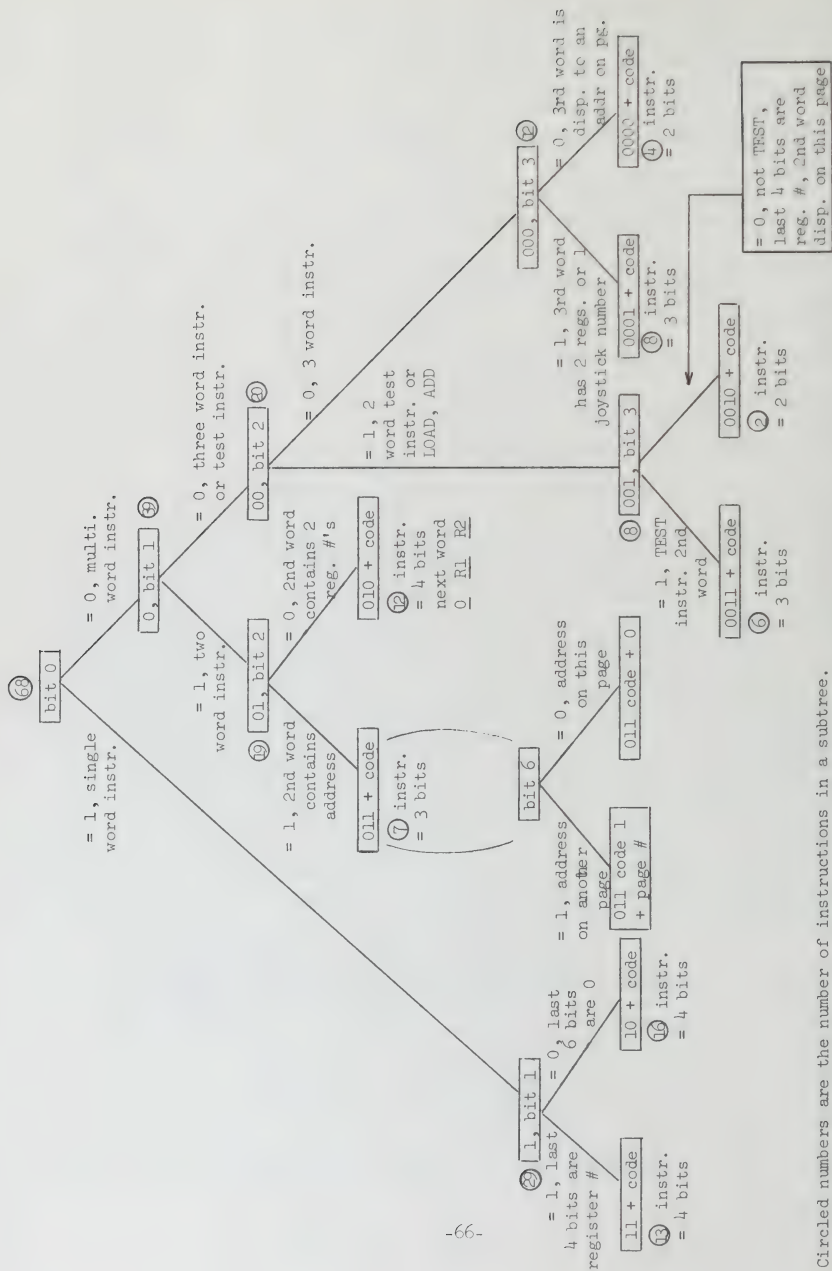


Figure 1.

Currently, the routines that can be used from the old system are being modified to run as subroutines called by the interpreter (see 3.1.6).

The interpretive language and the op codes are as follows:

INTERPRETER

System Basics

1. Registers 0-15

Each register contains a coordinate (X or Y). X,Y pairs can be referred to by the lower register (i.e., 2 refers to 2 and 3). The coordinates of the last joystick hit are in 0,1.

2. There is a vector for each console that contains the registers, line buffer, six character name buffer, and bits to indicate which segment hit and whether the 360 is there or not, and 32 test bits.

3. The current data structure contains all the information necessary to draw the current picture.

OP CODE

<u>bits</u>	<u>octal</u>		
100000	40	ERASE	erase the screen
100001	41	INIT	initialize current picture
100010	42	DISP	display current picture
011000	30	DISP NAME	display the picture 'NAME' but it is <u>not</u> put into the current picture structure
011001	31	FETCH:ADDR	} fetches the picture 'NAME' or the name in the name buffer. The picture is placed in the current picture structure. The location ADDR is branched to if the picture is not found
000000	00	FETCH NAME:ADDR	
011010	32	STORE NAME	} the current picture is stored under the name 'NAME' or the name in the name buffer
100011	43	STORE	
011011	33	CALL SUB	call subroutine 'SUB' (requires a parameter stack)
111100	74	DISPL N1	display text line in buffer at X,Y in registers N1, N1+1

0001011	054	TEXT 'XXX',NUM	put 'XXX' or ALPHA into line buffer (ALPHA must be the
0001111	074	TEXT ALPHA,NUM	address of a text string). NUM is optional and is the length of text desired in characters.
0001111	074	TEXTD N1,'XXX',NUM	put 'XXX' or ALPHA in line buffer and display at TEXTD N1,ALPHA,NUM/X,Y in registers N1, N1+1. (NUM same as above.)
0001000	220	LINE N1, N2	draw line from coordinates in N1, N1+1 to N2, N2+1
0001001	224	LINED N1, N2	same as above but also put into current data structure
0001010	230	DLINE N1, N2	delete line whose endpoints are in N1, N1+1 and N2, N2+1
0001000	240	CONN N1, N2	connect nodes with coordinates in N1, N1+1 and N2, N2+1 (visible connect)
0001001	244	ICONN N1, N2	invisible connection
0001010	250	UNCONN N1, N2	disconnect nodes whose coordinates are in N1, N1+1 and N2, N2+1
100000	60	CRTERM N1	create a terminal at coordinates N1, N1+1
100001	61	DTERM N1	delete terminal at N1, N1+1
000011	03	INST N1:ADDR	get instance whose name is in the name buffer and put up at coordinates in registers N1, N1+1. Failure to find the instance causes a branch to ADDR.
100010	62	DINST N1	delete instance whose coordinates are in N1, N1+1
100100	44	CRMN	create mnemonic structure from current picture
100101	45	SEND	send current structure to 360 (does not include line buffer)
0001000	260	SEND 0,1,...,N	send parts of structure to 360 (0 = header, 1 = line block,..., 11 = mnemonic block, N ≤ 15)
100110	46	SENDL	only send line buffer
100111	47	SENDH	send joystick hit coordinates and button number to 360
100000	50	REC	receive structure from 360
0000000	200	MOVE N1,N2	move register N2 to N1
000101	35	GOTO ALPHA	unconditional branch to ALPHA
100001	51	WAIT	wait until an input is received from 360, keyboard or joystick
100100	64	INSERT N1	insert line buffer at coordinates in N1, N1+1
100101	65	DELETE N1	delete line at coordinates in N1, N1+1
100100	52	APPEND	append line buffer to end of current block
100110	66	CHANGE N1	put line up to coordinates in N1, N1+1 in line buffer
0000000	100	LOAD N1,X	load register N1 with coordinate X

} Text Editor
Commands

TEST OPTION:ADDR

this will test to see whether the option has occurred.
If so, it will go to ADDR.

Option

JO11000	140	a. INPUT--is 360 data available?
JO01000	040	{ b. JOY1,..., JOY12--has joystick button 1,..., 12 been hit?
		{ c. JOY--any joystick hit?
JO11001	144	d. TEXT--a text line entered?
		e. R1, LOGICAL, R2: ADDR--register R1 is compared to R2. If the conditions specified by 'LOGICAL' are met, a branch to ADDR is made. Otherwise, the program will drop through. The possibilities for 'LOGICAL' are:
JO01001	044	GT--greater than
JO01101	064	GE--greater than or equal to
JO01100	060	EQ--equal to
JO01110	070	LE--less than or equal to
JO01010	050	LT--less than
JO11010	150	f. TERM: ADDR
JO11011	154	INST: ADDR
JO11100	160	END: ADDR
JO11101	164	BLOCK: ADDR

These options will allow the coordinates in O,l to be tested to see if they are within dX,dY (the "allowable miss" range) of a terminal, instance, end of a line, or block of text, respectively. If the hit is within the range, the actual coordinates of the terminal, instance, end of line, or block of text will be put in O,l and a branch to ADDR will be taken. Otherwise, it falls through with O,l undisturbed.

JO00001	01	g. NAME, ALPHA: ADDR
		NAME, 'XXX': ADDR

The name buffer is compared with ALPHA or 'XXX'. If equal, the branch to ADDR is made.

JO00010	02	h. BIT, NUM: ADDR
---------	----	-------------------

If bit NUM($0 \leq \text{NUM} \leq 31$) is on, a branch to ADDR is made.

SEGMENT

allows the screen to be segmented into 16 areas. The segments can be specified as follows: (All else is disabled)

1. X1, X2, Y1, Y2, ADDR

Giving the X and Y extremes of the area. The address ADDR will be branched to if there is a hit in the area.

2. X1, Y1, (DX, DY), ADDR

Center point with a DX and DY
Numbers are decimal unless surrounded by
0'7' which indicates octal.

011110	36	USEG ALPHA	puts the segment bounds specified in ALPHA into the console vector. Once this command has been executed, a joystick hit in any one of the segments will cause a branch to the appropriate routine specified in 'ALPHA SEGMENT'.
011111	37	CUSEG ALPHA	does a 'USEG' and then compares will be done to see if the coordinates in 0,1 is in one of these areas.
010111	53	MENTER	enter the name in the name buffer into the menu
010100	54	MDEL	delete from menu the name in the name buffer
010101	55	PMENU	put up previous menu segment
010110	56	NMENU	next menu segment displayed
		MENU X1,Y1,N1	X1,Y1 are the coordinates of the upper left corner of menu area. N1 is the number of lines in menu. A hit in this area will cause the hit name to be put in the name buffer.
		DCON 123 DCON 'XXX' DCON 0'123' }	Definition of a constant is made. A number is assumed to be decimal unless surrounded by 0'123', in which case it is octal. Any characters enclosed in single quotes will be compiled as a text string.
010111	67	SCAN N1	The scanner puts the character in position specified by N1 into name buffer. Then the next characters up to a maximum of six until a nonalphabetic character is scanned and put into the name buffer. If there is a mnemonic designation (.0), this is appended to the name.
0100001	204	MOVE2 N1,N2	moves two registers at a time, i.e. N2 to N1 and N2+1 to N1+1
010111	57	RETURN	causes a return from a subroutine call or return to monitor
		EXTERNAL ADDR	makes the name ADDR available as an entry point for an external subroutine call
		END	physical end of program
111000	70	SET A1	} sets bit A1 or bits A1 through A2 on
0100010	210	SET A1,A2	

111001	71 CLEAR A1	} clears bit A1 or bits A1 through A2
0100011	214 CLEAR A1,A2	
111010	72 EXT N1	make the terminal number in N1 external
111011	73 INT N1	make the terminal number in N1 internal
0010010	110 ADD N1,X	add constant X to register N1
0100111	234 ADDR N1,N2	add register N2 to N1 and put result in N1
111101	75 NTERM N1	put coordinates of the terminal whose number is in N1 into registers 0,1

3.1.6 Interpreter Language Assembler (J. Stynes)

This quarter an assembler was designed for the interpreter language described in section 3.1.5. The basic function of the assembler was to translate a source program into code which could be executed by an interpreter running on the PDP-8. The following is a description of the design to be used in implementing the assembler.

As the interpreter uses paging, the main criterion in designing the assembler was to try to keep all data on the same page that it was referenced, and thereby reduce the amount of swapping necessary. This was done by using two program counters--the TPC, which counts from the top of the page down; and the BPC, which counts from the bottom of the page up. The TPC points to the current instruction while the BPC keeps track of data, i.e., literals or instructions involving DCQN or SEGMENT statements. Thus, any time data is referenced by an instruction, the BPC is decremented the appropriate amount, the data is inserted into the page, and its starting address is stored in the instruction. (Naturally, before each instruction and its data are inserted, the amount of space required is checked against the amount of space available, BPC-TPC. If there is not enough space available, a new page is started and the TPC and BPC are initialized.)

In implementing the above scheme of two location counters, there are two important considerations:

1. In order to avoid a third pass by the assembler, all variable length data must be defined before it is used. (Variable length data appears in SEGMENT statement, and data which is referenced by the TEXT instruction.)

2.. Since data must appear on the page it is referenced, there is a possibility that the same data may appear on several different pages. Thus, there must be a copy of it somewhere in core so that it can be passed to the page which requires it. This means that each `DCON` or `SEGMENT` statement that is encountered in the source program is decoded and stored in a constants area, and its size and address are stored in the symbol table under the label given to the statement.

The assembler is a basic two-pass assembler. In the first pass the symbol table is constructed, while in the second pass the statements are decoded and all addresses are resolved. In pass one, it is only necessary to scan as far as the operation code to determine what to increment the BPC and TPC. (Except in the cases of the three instructions which have variable length data. In this case, the operand is scanned and the length is gotten from the symbol table.) After scanning a line in pass one, the assembler copies the source line, and the information it has scanned so far, onto a temporary data set. In pass two, the line is read back in, and after doing whatever particular processing is necessary for a given instruction, any one or all of the following three routines are called to scan off and pack the rest of the line:

1. `REG(WORD,M)`: assumes that the next item to be scanned is a register. If any errors occur, the appropriate error flags are set. If not, the register is packed into `WORD` starting at bit `M`, packing from right to left. For example, if `M = 4`, the register would be packed into bits 4, 5, 6, 7 with 7 the high-order bit.

2. `ADDR(WORD)`: assumes that the next item to be scanned is an address. If any errors occur, the appropriate error flags are set.

If not, the address is found in the symbol table, the displacement of the address from the top of the page is packed into WORD, and the page number is packed into the five low-order bits of the word preceding WORD.

3. NAME(WORD): assumes that the next item is a reference to a constant. Error checking is done, as usual. If no errors occur, the program determines the type of constant, stores it in the appropriate position in the page, resets the BPC, and packs the displacement of the constant from the top of the page into WORD.

After each line is processed in pass two, it is printed out along with its octal representation and the statement number. General flowcharts of passes one and two are listed in the following flowchart.

In the 1st Quarterly Progress Report 1973 a detailed description of the assembler will be presented.

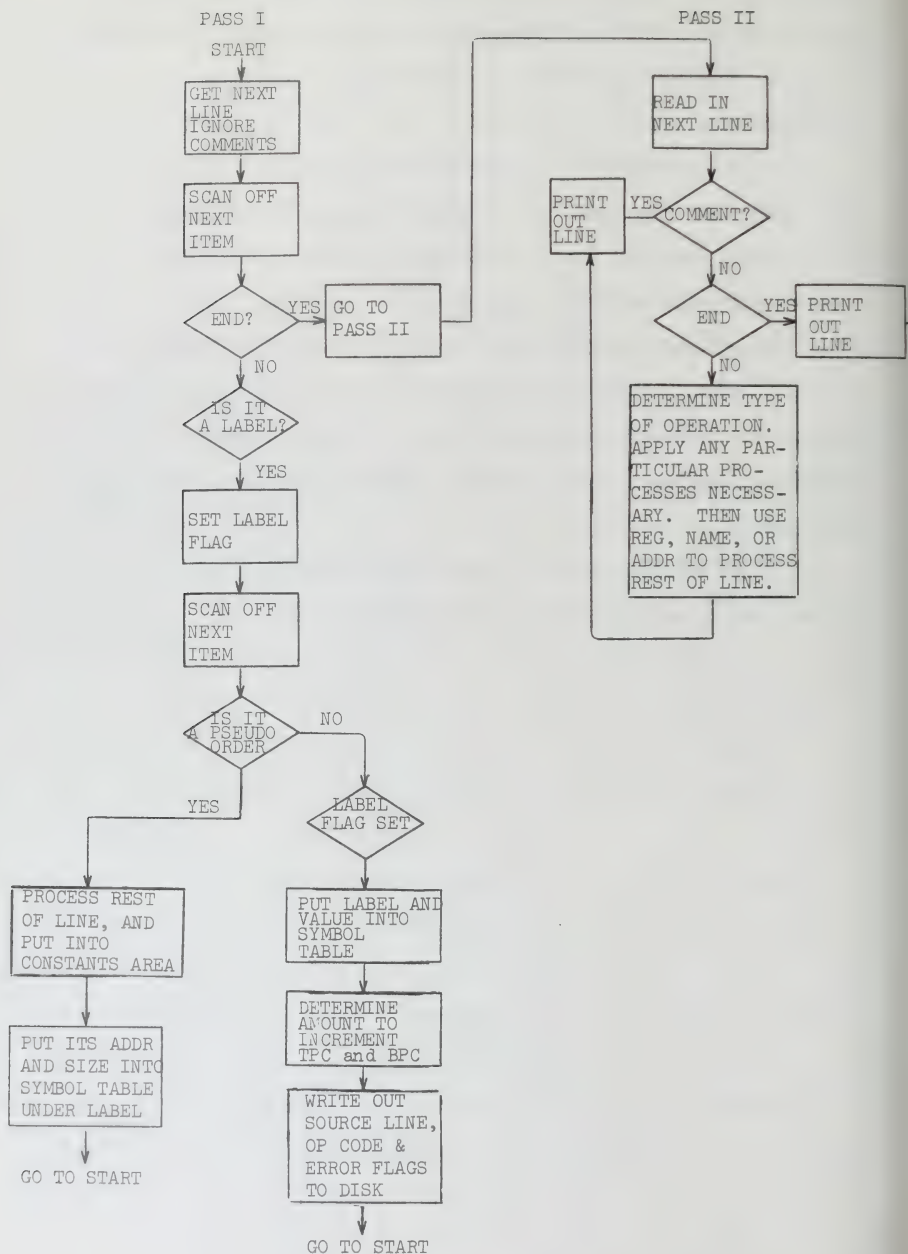


Figure 2.

3.2 Illinois Graphics Computing System (IGCS)

3.2.1 Overlay Supervisor (D. Mueller)

The final version of the supervisor was completed. A modification was included to allow monitors access to the last system control and read. The programmers manual for the supervisor is in the process of being typed and will be available shortly.

3.2.2 A Drafting Language (ADL) (T. Runge)

A final version of the ADL compiler was implemented and will soon be made into an overlay ready for use in the final system. A first approximation to a user's manual for ADL is on file with Mike Pleck, Department of General Engineering.

3.2.3 Plot Sectorization Overlay (T. Runge)

The routines which will comprise the sectorization overlay appear to be in their final versions although complete debugging of sectorization is difficult except as a part of the complete system. Specifically, the sectorization buffer handler, GETBUF, and the sectorization monitor were completed this quarter as well as the character entry routine, CHRENT, which converts compiler output for text strings into a form acceptable by the plotting phase.

3.2.4 ILLISM-E (W. Tam)

A macro assembler (MACRO-11) became available on the PDP-11. This contained an extension of the PAL assembly language plus macro capabilities. The ILLISM-E compiler was modified considerably to take advantage of the macro facilities. This is in accordance with one of

the primary goals of the compiler, i.e., a macro-based system for easy implementation.

Due to the extra work involved in pulling user-specified arithmetic functions from the disk library, and then properly linking and relocating them, the current version has a fixed set of arithmetic functions which the user can use. This set of functions is already linked to the compiler and can either be core resident or in an overlay. Also, a user's manual is being prepared to describe the operation of the compiler.

The numerical values of the variables during actual simulation of models are now written onto a disk file at specified intervals (time-steps). This file then becomes available at the termination of the simulation. It can either be dumped onto the line printer or be retained as an input file to a graphics routine for plotting.

3.3 Graphical Remote Access Support System (GRASS)

3.3.1 Implementation of GPSS for the Simulation and Modelling System (B. Purvy)

It is planned to adopt IBM's General Purpose Simulation System to the GRASS system in order to provide a means of simulating discrete systems which are already in general use. The user will be able to draw a flowchart of his system on a GRASS terminal and send it to the remote computer where it will be converted into standard source language for the GPSS assembler. Some interaction will be possible, possibly allowing the user to correct errors in his program and rerun it, look at his output, and change relevant parameters, etc. Graphic output from the completed simulation run will be possible, either from the standard graph module provided with GPSS, or with a more generalized plotting program. Provisions may be made for the user to look at the output generated by GPSS, and choose which parts he wants hard copy for.

A tentative program for drawing flowcharts with the GRASS system was written in the interpreter language which is currently being implemented. The program is currently for a general FORTRAN-type flowchart with four block types and various options allowing the user to accept the default movement across the screen or specify which way he wants to proceed. (See Figure 3.)



Figure 3. Menu Area for Flowchart Program

The basic blocks for programming are the square and the diamond. The circle represents a reference to another "page." The user will draw on the screen until he wishes to save the current picture and continue on a blank screen; he will then hit "PAGE." He will be asked to give a name for the picture. The circle with the arrow on top is a page entry symbol, the other end of the external connection. It will have information linking it to the appropriate circle on a previous page. The four arrows allow the user to specify in which direction the next block will be in relation to the block just drawn. If no arrow is hit, the next block will be put up in the same direction as the last move, i.e., if he wishes to go straight down (the default) or to the right for several blocks, he need only indicate that once. The terminals between blocks are automatically

connected for the user unless he has hit "REORG"; in this case he begins a new line of blocks. This allows for the case frequently seen in GPSS where tasks are generated independently of each other. The DRAW and CONNECT(I) allow for manual connection of terminals: this would be used for blocks not adjacent to each other; the user might connect then invisibly and draw a broken line representing the connection.

Work has begun on the 360 program to turn the completed flowchart into source code. The user will enter the statement label--if any, block name, and operands as parameters for each instance. The GRASS software currently in use does not display parameters as part of the picture; this will be modified in the new system. Statement labels will be generated for blocks referenced by other blocks by means of terminal connections. Thus, a program can be specified either logically--leaving the work of creating labels and inserting them in the operands to the system, or written as straight line code--with all statement labels already there.

3.3.2 Monitors (L. Lopez)

MYFIOCS#

As part of the conversion of CIRPAC, a circuit analysis package, modifications of MYFIOCS# were found to be necessary.

FIOCS# is an IBM supplied subprogram. FIOCS# stands for FORTRAN Input Output Control System. The '#' is used to prevent accidental calls to FIOCS# by FORTRAN programmers.

MYFIOCS# is a replacement for FIOCS# for use in the PDP-8 graphics display environment. Its essential work is to provide special

services for units 1 and 2 when used by FORTRAN I/O statements.

It also provides rerouting of error messages to both the printer and the display, when such messages are produced by the other components of the system.

The major changes to MYFIOCS# were to implement unformatted I/O and the control statement REWIND.

IBCOM# is a multiple entry point subroutine called by FORTRAN programs to perform I/O services. IBCOM# calls MYFIOCS# at entry point FIOCS#. FIOCS# performs the actual calls to the operating system routines.

MYFIOCS# when reading on unit 1 and writing on unit 2, communicates with the PDP-8 graphics system instead of with the usual O/S files. MYFIOCS# supports only fixed length records. Hence, the FORTRAN default, variable spanned records, is not compatible with it.

The calling sequence of FIOCS# is shown in Table 1.

For read statements, FIOCS# is called once with M1 = 0. FIOCS# performs initialization, reads a record and returns. If IBCOM# determines that two or more records are required, then additional calls to FIOCS# with M2 = 1 will occur. FIOCS# would then perform additional reads.

For write statements, FIOCS# supplies only a blanked buffer when called with M1 = 0. Subsequent calls (M1 = 2) cause FIOCS# to print the buffer and blank it again.

The control statements (M1 = 3) are handled in a straightforward manner.

The close command (M1 = 4) is used at job termination, and causes the closing of all files.

Status of all files, except 1, 2, 5 and 6, is contained in UNITBL, the Unit Table.

Useful information about MYFIOCS# is given below in Tables 1 through 4. Some minor FIO99Z code has not been specified in Table 4.

```

LA      R2,PARM
L        R1,=V(FIOCS#)
BALR    R0,R1
DC      AL1(M1),AL1(M2)
B        ERROR

```

Table 1. FIOCS# Calling Sequence

<u>M1</u>	<u>M2</u>	<u>NOTE</u>	<u>USE</u>	<u>LABEL</u>
0	X'FO'	1,2,6	FORMATTED READ	FINIT
0	X'FF'	1,2,6	FORMATTED WRITE	FINIT
0	X'OO'	1,2,6	UNFORMATTED READ	FINIT
0	X'OF'	1,2,6	UNFORMATTED WRITE	FINIT
1	0	6	READ ANOTHER RECORD	FREAD
2	0	5,6	WRITE RECORD	FWRITE
3	0	3,4	BACKSPACE FILE	FCTRL
3	1	3	REWIND FILE	FCTRL
3	2	3,4	END FILE	FCTRL
4	0		CLOSE ALL DATA SETS	FCLOS

Table 2. Values of M1, M2

NOTES:

1. FORTRAN parmlist used as PARM. SAVEERR, SAVEEND and UNIT are set from parmlist.
2. READ causes a read and returns a buffer to IBCOM#. WRITE returns a blanked buffer to IBCOM#.
3. First word of FORTRAN parmlist is only one that occurs.
4. Not implemented.
5. PARM is length to be written.
6. Buffer is returned.

FGW Branches to FGWW if O/S file.

(FGW1) Otherwise, returns to IBCOM# with new FIO99Z buffer.

(FGWW) Does a write on the unit. Returns buffer to IBCOM#.

FREAD Branch to BREAD1 if unit 5.

 Branch to SBREAD if unit 1.

 Branch to BRET if O/S.

 Branch to FGW1 to get buffer.

FCTRL (M1 = 3) CONTROL STATEMENT.

 Prints diagnostics for backspace or end file

 and terminates.

 For rewind file is closed and a return to

 IBCOM# is made.

GETUNIT Subroutine to get address of unit entry in UNITBL

 from FORTRAN parameter list.

 R2 points to parmlist.

 R6 is RETURN address.

 R8 is (UNIT ENTRY) address, UNITBL for this unit.

FCLOS (M1 = 4)

 All data sets are closed. Return is made to IBCOM#.

CLS R8 points to a unit table entry. If unit is open

 for INPUT or OUTPUT, it is closed, R0,R1,R2,R7,R8,R10

 destroyed. R6 is RETURN address.

CLOSE R0,R1,R10,R7 destroyed.

 R1 is DCB address.

 R2 is BUFFER address.

 DCB is closed.

 Space for DCB and BUFFER is released.

 R7 is RETURN address.

OPN Opens non-special unit for input or output. R5 is set to DCB address. R6 is set to RECORD address. R8 should be set to an open list. R7 is RETURN address.

BRITE Write initialization (ML = 0) branch to SWRET if unit 0. Unit 0 is taken to mean the unit(s) where error messages are to be written.

 Branch to SWRET if unit 2.

 Branch to GW if not unit 6.

(WRET) Blank buffer. Return buffer and length to IBCOM#.

RET Reloads registers. Stores buffer address and length in two word area in IBCOM#, and then returns to IBCOM#.

MYERR Branches to ERRSAVE if specified. Branches to GOAWAY if unit 1 or 2. Otherwise, terminates with diagnostic.

 MYERR is specified as the SYNAD exit of all FIOCS# files.

GW Opens file for output if necessary. Blanks buffer return length and address to IBCOM#.

MYEND Same as MYERR but for ERRSAVE.

FRITE (ML = 3) WRITE RECORD.

 Branches to WALL if unit 0, error message.

 Branches to WCON if not unit 6.

 Writes buffer to printer.

 Branches to WRET.

WALL Writes BUFFER TO PRINTER. BRANCH TO WCOM.

WCON Branches to FGW if not unit 2.

WCOM (used by WALL)

 Writes BUFFER TO PDP-8

 Return buffer address and length to IBCOM#.

DC	ALL(FLAG),AL3(UNIT)
DC	A(FORMAT) ADDRESS OF FORMAT
DC	A(END) END=
DC	A(ERR) ERR=

Table 3. FORTRAN Parmlist

First word is required. All optional words are missing if not used.

FLAG

X'01'	UNIT IS ADDRESS OF UNIT #
X'04'	SYSTEM MESSAGE
X'10'	END= SPECIFIED
X'20'	ERR= SPECIFIED

Table 4. Interesting Labels in FIOCS#

ENTRY	Branch table for M1.
FINIT	Initialization routine. (M1 = 0). Store data set reference number in UNIT.
(OPTIONS)	Sets ERRSAVE and ENDSAVE to values specified by "ERR=" and "END=". Branches to BREAD if READ, or BRITE if WRITE.
BREAD	Branches to SBREAD if unit 1. Branches to GR if not unit 5. Opens card file (if closed). Reads a record. Returns buffer and length to IBCOM#.
SBREAD	Reads a record from PDP-8 system, translates it to EBCDIC, and returns buffer and length to IBCOM#.
GR	Non-special unit. Opens file if closed. Read RECORD. Returns buffer and length to IBCOM#.

GOAWAY	Retry procedure for PDP-8 I/O.		
FI099Z	Routine to associate array with file.		
FI099X	Disable FI099Z.		
UNITBL	Unit table. 99 16-word block containing		
	status of files. Format follows:		
UNE	DSECT		UNIT TABLE ENTRY.
UFLG	DS	X	FLAGS.
URD	EQU	X'80'	IF INPUT FILE
UWR	EQU	X'40'	IF OUTPUT FILE
UBUF	EQU	X'20'	IF FI099Z FILE
UPPOINT	DS	OAL3	--> CURRENT RECORD IN BUFFER.
URDCB	DS	AL3	ADR OF READ DCB.
URECL	DS	OA	LRCL FOR FI099Z I/O.
UWDCB	DS	A	ADR OF WRITE DCB.
UBUFR	DS	OA	BUF ADR FOR FI099Z I/O.
UWBUF	DS	A	ADR OF WRITE BUFFER.
UHIGH	DS	OA	ENDING ADR + 1 FOR FI099Z I/O.
URBUF	DS	A	ADR OF READ BUF.
UNEL	EQU	*-UNE	
UNIT	Unit number of file, saved by FINIT.		

ENDSAVE,ERRSAVE

End of device address and synchronous error exit

address saved by FINIT. Zero if not present.

3.3.3 FETCH Routine (A. Whaley)

The 360 facility for loading programs from a standard program library does not permit preassigned storage locations for the loading process. The swapping system requires that programs be loaded into the swap area. The OS/360 fetch routine IEWFELCS (version of IEWFETCH, which supports LCS) was modified for use as a subroutine to be called by SPACT. A new version of the GETMAIN macro is available, called GETUM, which calls a resident subroutine for storage allocation instead of making requests to OS. In addition to being at least 50 times faster, this routine allocates storage from the swap area only. FETCH uses GETUM to obtain storage for programs.

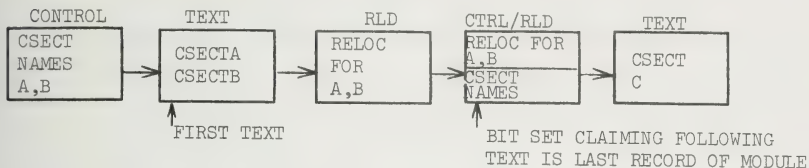
Modification of FETCH was performed which was not immediately necessary, but was felt to be advisable. The original version uses appendages for I/O which violate all rules for appendages and for good programming. Also, it was not desired that certain parts of the GRASS system be executed in supervision state if unnecessary. When the program was rewritten as a normal subroutine and unused code and code-in-error was removed, the program required less than half the original amount of storage.

In order to give a proper description of the workings of this program to the luckless souls who follow, it is first necessary to obtain a usable description of the format of the information as stored by the linkage editor. Preliminary information may be obtained by looking in the IBM System/360 Operating System, MVT Supervisor, Publication Y28-6659, Section 4, Contents Supervision, under the heading Fetching Routines and Modules to Main Storage; and in Section 12, Control Blocks and Tables under the headings Contents Directory Element to Entry Table inclusive. In the Release 18 manual, this information is around pages 112 and 308. The IBM Linkage Editor Program Logic Manual Y28-6610 has a variety of information. Of special interest to those working on FETCH is control/RLD records as listed in the index.

As a crude description of this program before I got it exists in the first reference (section 4), I will concentrate on basics and changes. The linkage editor may get a typical load module like so:

	START	LENGTH
CSECTA	00	4C8
CSECTB	04C8	2E
CSECTC	0528	1C00

For this load module, typical linkage editor output would be:



Among other goodies in partition data set directory entry for such module, is the disk address of the first text record, allowing FETCH to bypass the first control record. FETCH sets up a channel program like so:

- 1 read TEXT
- 2 read count, multiple track mode
- 3 read RLD
Control, or
RLD/Control
- 4 read count, multiple track mode
- 5 no-operation

In the original version, step 5 was modified while the channel program was executing to transfer control to another channel program looking just like this. If a text record was to follow, step 5 transferred to step 1 of the following channel program, otherwise to step 3. A three buffer, three channel program circulating system was used to read and process all information without losing disk revolutions. Because of the overwhelmingly cumbersome and poorly implemented techniques used, this system has been dropped.

Currently, the channel program is executed (steps 1 to 4) and then terminated. If the second record is an RLD or RLD/Control record, then relocation is applied to all elements listed in the record. When the last text record is to be read, the channel program is modified to terminate after the read (if no RLD records follow--determined from the preceding control record).

LIST OF PUBLICATIONS

- Des Trois Maisons, P., "Some Proposals for Modifying the Grass Graphics Language," UIUCDCS-R-72-554 (December 1972); submitted as Master's thesis, February 1973.
- Gear, C. W., "A Generalized Interactive Network Analysis and Simulation System," USA-Japan Computer Conference, Tokyo, Japan, October 3-5, 1972.
- Gear, C. W., "The Effect of Variable Mesh Size on the Stability of Multistep Methods," Conference on Ordinary Differential Equations, Austin, Texas, October 19-20, 1972.
- Larsen, L., "Automatic Solutions of Partial Differential Equations," UIUCDCS-R-72-546 (October 1972); submitted as Ph.D. thesis, October 1972.

U.S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

AEC REPORT NO.

COO-1469-0218

2. TITLE

4th Quarterly Progress Report 1972

TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

- ☐ c. Other (Specify) _____

RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

REASON FOR RECOMMENDED RESTRICTIONS:

SUBMITTED BY: NAME AND POSITION (Please print or type)

C. W. Gear

Professor and Principal Investigator

Organization

Department of Computer Science

University of Illinois

Urbana, Illinois 61801

Signature

Charles W. Gear

Date

December 31, 1972

FOR AEC USE ONLY

AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION
RECOMMENDATION:

PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

4. IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH: ILLIAC III

4.1 Semantic Model Interpretation Methods for Image Processing (S. R. Ray)

The long range objective of our work is the development of semantic model interpretation methods for image processing. Although an example of the basic methodology had been developed and applied to the example of landscape scene interpretation, the method is strong in heuristics and weak in theoretical framework. We are examining an additional class of data (breast thermograms) with a view to development of a theoretical description of the semantic modeling method.

This period was devoted to preparation of image preprocessing programs necessary for the extraction of features from anterior-posterior breast views.

1. A flow diagram for this purpose was prepared and programmed.
2. Some images were obtained, digitized, and loaded into the 360/75 system. (We found the transmission and data handling problem to be extremely awkward.)
3. The programming tests reached the stage of ability to locate breast outlines--a preliminary step to defining the region of the picture on which features are to be extracted.

When the stage of reliably extracting features is reached, we expect to write a semantic model containing the relationships between image features which are believed to be predictive of cancer, and to study the theory of the modeling in more detail.

4.2 Covering Theory (R. S. Michalski)

Work was done on the investigation of formal properties of interval complexes, cartesian complexes, and cyclic complexes. A paper which presents results is being prepared.

4.3 Variable-Valued Logic (R. S. Michalski)

Program AQVAL/1, which synthesizes quasi-minimal VL_1 formulas, was applied to a number of examples for testing purposes. One of the examples was the problem of theory formation sent to us by Mr. Sridharan of Stanford University. The program produces a very simple VL_1 formula characterizing the input data.

A paper is being prepared on the theoretical properties of the VL_1 system.

4.4 Engineering and Maintenance (J. E. Robertson)

Initial steps were taken during the quarter toward achieving the ultimate goal of making the PAU available to ILLIAC III users. These include:

1. Static tests of stalactite cards of eight rows (25%) of the Iterative Array were made. Approximately 40% were replaced. Test and repair facilities were reactivated and construction of a few additional spares was begun.
2. A program (PAU 1) was written to enable information transfers into and out of the Iterative Array under PDF8/e control. Information transfer is on a row-by-row basis via the north border of the Iterative Array and the exchange net.
3. Controls for three new PAU instructions were designed and fabricated. These are
 - i) Echo Border REGISTER (EBREG)
 - ii) Load NORTH Border (LNORB)
 - iii) Store NORTH Border (SNORB)

The EBREG instruction is primarily for maintenance purposes and enables checking of the buffer register between the exchange net and the Iterative Array. At the end of the quarter, the EBREG instruction had been checked, including a special test incorporated in PAU 1. Check-out of the LNOB and SNORB instructions, and associated debugging of PAU 1, were completed by the end of the quarter.

PUBLICATIONS

- Borovec, R. T. and Lewis, G. T., "IMAGE 8: MAINTENANCE PROCESSOR/PAU INTERFACE," UIUCDCS-F-72-873, Department of Computer Science, University of Illinois, Urbana, Illinois, November 1972.
- Maruyama, K., "A STUDY OF VISUAL SHAPE PERCEPTION," UIUCDCS-R-72-533, Department of Computer Science, University of Illinois, Urbana, Illinois, October 1972 (Ph.D. Thesis).
- Read, J. S., "PARALLEL IMAGE-PROCESSING FOR AUTOMATED CERVICAL SMEAR ANALYSIS," UIUCDCS-R-72-497, Department of Computer Science, University of Illinois, Urbana, Illinois, October 1972 (Ph.D. Thesis).

U.S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

AEC REPORT NO. COO-2118-0044	2. TITLE 4th QUARTERLY PROGRESS REPORT 1972
---------------------------------	--

TYPE OF DOCUMENT (Check one):

☒ a. Scientific and technical report

☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

☐ c. Other (Specify) _____

RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

☒ a. AEC's normal announcement and distribution procedures may be followed.

☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.

☐ c. Make no announcement or distribution.

REASON FOR RECOMMENDED RESTRICTIONS:

SUBMITTED BY: NAME AND POSITION (Please print or type)

J. E. Robertson

Professor and Acting Principal Investigator

Organization

Department of Computer Science

University of Illinois

Urbana, Illinois 61801

Signature 	Date December 31, 1972
--	---------------------------

FOR AEC USE ONLY

AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION
RECOMMENDATION:

PATENT CLEARANCE:

☐ a. AEC patent clearance has been granted by responsible AEC patent group.

☐ b. Report has been sent to responsible AEC patent group for clearance.

☐ c. Patent clearance not required.

5. THEORY OF DIGITAL COMPUTER ARITHMETIC

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 813.)

5.1 Continued Fraction Arithmetic

A report summarizing the work done towards obtaining a special class of continued fraction algorithms for function evaluation and integration is in progress. These algorithms are based on applying a sequence of bilinear transformations to a Riccatti differential equation.* It may be recalled that a Riccatti equation is a nonlinear first order equation of the type $y' = ay^2 + by + c$.

(Kishor S. Trivedi)

5.2 Automatic Computation of Certain Elementary Functions Using Microprogramming

During this quarter, the previous results on this subject were carefully analyzed, and a report is being prepared.

(Peter D. Ting)

5.3 Function Evaluation Techniques

During this quarter the study of possible refinements of continued product algorithms^{1,2} has revealed that the original algorithm for logarithm evaluation can be speeded up by factor of 2 using a simple

* See Quarterly Technical Progress Report, APR-JUNE 1972, Page 178.

¹ B.G. DeLugish, "A class of algorithms for automatic evaluation of certain functions in a binary computer," Report No. 399, DCS, University of Illinois, Urbana, June 1970.

² M.D. Ercegovic, "Radix 16 division, multiplication, logarithmic and exponential algorithms based on continued product representations," Report No. 541, DCS, University of Illinois, Urbana, August 1972.

property that the scaled remainder R_k for $k \leq p = \frac{m}{2} + 1$, where m is the number of digits in the result, contains all information about the error committed by truncating the iterative process after the k^{th} step. For example, in the radix 16 case, we can have $\log X_0 \approx \log 16^{-p+1} \sum_{i=0}^p (\log M_i)$ instead of $\log X_0 \approx - \sum_{i=0}^m (\log M_i)$. Although the algorithm for evaluation of exponentials uses the same set of precomputed constants $\{\log M_i\}$, truncating the iterative process after the p^{th} step will require one extra multiplication, cancelling the possible speed-up. However, for the sequential schemes, where the result evaluation is performed after the normalization has been completed, the speed improvement by a factor of 2 is also possible.

A general approach of decomposing the process of function evaluation based on continued products into more than two partially dependent subprocesses, easy to implement, did not yield satisfactory results. Thus we have started investigating feasibility of some other approaches in function evaluation. This includes rearranged recursions using Chebyshev's approximating polynomials which are characterized by a small number of precomputed constants per function. Furthermore, no selection rules are required and the whole process of evaluation, being essentially a sequence of additions, can be conveniently speeded up. Also, some insight has been gained for an approach where the function is evaluated by summing precomputed constants selected on the basis of k digits of the argument.

(M. D. Ercegovac)

6. SWITCHING THEORY AND LOGICAL DESIGN

(Supported in part by the National Science Foundation under Grant Number U.S. NSF-GJ-503A #1.)

Transformation techniques to improve given networks were continued to be tested by computer during this period. Major efforts were concentrated on the preparation of writings on the transportation and wired-logic.

S. MUROGA

During this quarter, our main task was the documentation of the procedures and corresponding computer programs which have been developed for the transformation of non-optimal NOR-gate networks into "nearly optimal" networks. Many "bugs" and unnecessary or inefficient computations were discovered during the documentation process. These mistakes have been corrected or noted for later correction. Many of the DCL Reports (containing the documentation) have been more or less finished, awaiting only final rewriting and the inclusion of computational examples.

Optimal (number of gates minimized primarily, number of connections secondarily) AND-OR-gate networks (with both complemented and uncomplemented external variables available) for 219 (out of a possible 222) NPN-equivalence class representative functions of 4 variables have been found. This duplicates a previous effort whose results were discarded when an error was discovered in the program used to obtain the results (ILLOD - (ANDOR-B)). Each of the 3 remaining representative functions has been run for more than 3 hours without termination of the computation. Thus optimal networks for the 3 corresponding equivalence classes are still unknown.

(J. N. Culliney)

A number of interesting experiments have been run on the computer using the network transformation programs developed. The logic design system used in these experiments consists of the following two parts:

- (a) Synthesize a logic network which realizes a given function but which may have many redundant gates and connections; and
- (b) Apply the network transformation procedure using error compensation.

The truth tables for the 30 five variable test functions used in these experiments were obtained using a random number table. In part (a) the logic networks were obtained from the following three sources: the first solution of the branch-and-bound method, a Universal network, and a simple procedure to synthesize a three level network. The single path method was used to apply the network transformation procedure in part (b). Table 1 shows part of the results of applying the above design system to the 30 five variable functions.

Function Number	Initial Network Cost ⁽¹⁾⁽²⁾	Final Network Cost	Initial Network Cost ⁽³⁾	Final Network Cost	Initial Network Cost ⁽⁴⁾	Final Network Cost
1	19073	12039	33305	13040	26102	14045
2	20069	14045	33305	13044	25100	14042
6	14043	9027	33315	8029	18081	8029
7	15052	13036	33305	12041	25100	11037
19	16054	9025	33308	9025	26104	9025
27	13038	11031	33307	11036	23097	12033

TABLE 1. Results by the Network Transformation Procedure

- (1) The cost is represented as $1000 \times (\text{number of gates}) + (\text{number of connections})$
- (2) The first solution by branch and bound
- (3) A Universal network
- (4) A three level network

Several interesting observations can be made from the results in Table 1. First, the three initial networks for each function are quite different. Second, the application of the network transformation procedure does not consistently give better results for one particular kind of initial network. For example, the best results for function 1 is obtained when the initial network is the first solution of the branch and bound method while the universal network results in the best network for function 2; and the three level network results in the lowest cost final network for function 7. Thus it is not possible to determine before running the program which

initial network to use for a given function.

Each of the 30 five variable functions was also run for 20 minutes using the branch-and-bound program only. The best result for each function by the transformation procedure is as good as or better than the result by branch-and-bound only except for function 27. Branch-and-bound obtains a network of cost 9028 for function 27. However, it is very important to note that the longest calculation time required to obtain any of the results by the transformation procedure was 23.05 seconds and that the average calculation time was 7.18 seconds. Also the branch-and-bound program was used to find the optimum solution for functions 6 and 19. Function 6 has an optimal cost of 8029 and required 2584.23 seconds of computation time while function 19 has an optimal cost of 9025 and required 3912.09 seconds of computation time.

A number of other experiments are currently being run. These include the use of the multi-path method of applying the network transformation procedure. Also the network transformation procedure program consists of a number of subroutines which can be called in various orders. Experiments are under way to try to determine the best design system.

(K. Hohulin)

(1) Properties of networks using wired-logic were considered. Wired-logic refers to the capability of typing the outputs of two or more gates together to perform additional logic without any extra components. Since wired-logic has no associated cost, the usage of wired-logic will reduce the cost of a network. Very few works, however, have considered properties of a network with wired-logic. Most logic gates which are implemented with

bipolar transistors are capable of accommodating wired-logic:

- (a) A combination of NOR gates and wired-ANDs (it is equivalent to a combination of NAND gates and wired ORs) such as RTL and Injection logic.
- (b) A combination of NOR gates and wired ORs (it is equivalent to a combination of NAND gates and wired ANDs) such as DTL, HTL, TTL, and ECL. Properties of networks using NOR gates and wired ANDs or NOR gates and wired ORs were considered and synthesis procedures for optimum networks were developed.

(2) Our logic design system of NOR networks consists of many network transformation procedures. Many combinations of procedures were considered to make efficient systems.

(3) Descriptions of procedures developed were also prepared in this period.

(Y. Kambayashi)

The entire period was devoted to debugging and documenting the NOR network transformation procedures and corresponding programs developed during the past year. Several programs applying different transformation procedures to different initial networks were developed and tested. During the documenting and testing process several errors in the NOR network transformation by error compensation program were found and corrected. Some redundancy in a couple of transformation procedures was also discovered and eliminated. This will improve the efficiency of the programs to some extent.

(H. C. Lai)

7. MACHINE AND SOFTWARE ORGANIZATION STUDIES

(Supported in part by the National Science Foundation under Grant Number
US NSF GJ 27446)

The following is a collection of related work aimed at improved designs for computer hardware and software systems. We are interested in parallel and pipeline processors, data alignment networks, small primary memories, effective use of rotating memories, and some questions concerning user languages for problems including typical FORTRAN type calculations. Our approach is to attempt to study classes of algorithms theoretically as well as to measure various parameters of existing programs. We are also studying a variety of file processing problems including the analysis and design of special purpose machines and their software.

(D. Kuck)

7.1 FORTRAN Parallelism Detection - (D. Romine, R. Strebendt, and R. Towle)

During October we worked on scanning and analyzing programs from various sources, e.g., numerical analysis books, non-linear optimization book, programs from waste can, etc. We then put together a program to print out histograms for the combined data (SCATPLT), and wrote a program to read in and accumulate these data sets into a conglomerate data set. We then reran all statistics of all programs previously analyzed.

Three different experiments were run this quarter. First, the FORTRAN decks were analyzed with all reasonable variations. The results are still being processed. Second, timing studies of the analyzer were performed to determine the speed of the analyzer and the bottlenecks. The speed of the analyzer is now about 2 cards per second, which is a speedup of 2 over last January, though the analyzer is now executing out of slow core versus fast core then. Many of the bottlenecks are due to PL/1's handling of arrays.

Several assembler routines have been written to help this problem.

Third, experiments have been made on BAS and IF trees. From these experiments we plan to consider machine organizations for BAS evaluation, comparison of algorithms for tree-height reduction, and classification of FORTRAN BAS's. Hopefully, by the end of January, the raw data from the BAS experiments will be processed.

At the same time these three experiments were made, a transition to a private disk pack was performed. Also, a User's Guide to the Analyzer was written, and the Analyzer was sent to two potential users.

During the past quarter the capability of generating statistical summaries over an ensemble of programs was implemented.

The new program prints the following information:

- 1) Tables showing the common incidence of various parameters, such as speedup and the number of processors (e.g., the number of times a particular speedup value was associated with a particular number of processors).

- 2) Histograms of various parameters and measures by the number of program traces for which each parameter value was found.

- 3) Histograms of various parameters and measures by the number of programs for which each parameter value was the average value found in that program.

- 4) A table describing the characteristics (such as the number of blocks of assignment statements found within DO loops) for the ensemble.

- 5) A table giving the maximum, minimum, and average of a number of parameters for the ensemble.

The program is capable of generating either high-speed printer plots or CALCOMP plots of the histograms.

7.2 DO Loop Analysis - (S. C. Chen)

Since the various algorithms for detecting DO loop parallelism have been implemented and tested extensively, the investigation of analytical properties of DO loops and the theoretical bounds of parallelism within any loop structure was begun. To facilitate this analysis, some basic assumptions of loop structures have been made. Preliminary results show that a lower bound on speedup may be obtained by using only a few parameters which can be easily established from any given loop structure. Theorems for some simple loops were formed in a unified way. During the next quarter, a general class of loops with sharper bounds, and more parallel properties of loops will be under investigation.

7.3 Memory-Processor Connection Networks - (D. Lawrie)

In order to utilize the potential speed of a SIMD type parallel processor it is necessary to arrange data in the memory system so that subsets of this data can be fetched in parallel without memory conflicts. Additionally, we must provide sufficient memory-processor paths to allow the data to be correctly aligned with the processor array. We have investigated several storage mapping algorithms together with a memory-processor interconnection network, and have demonstrated the cost effectiveness of these and compare them with other networks which have been proposed for this application.

7.4 D-Machine Microprogram - (J. Rinewalt)

The D-Machine was down during most of this quarter due to a power supply failure last summer. When the power supply was replaced, the printer interface was checked out and a microprogram for listing cards on the printer was written and debugged.

The file processing microprogram should be checked out by the end of January. Since the word and string instructions were debugged via the

Simulator, no major problems are expected. The I/O instructions, however, could not be simulated and will probably require extensive testing. Check out of the S-Language program should begin in early February.

7.3 Information Retrieval Computers - (W. Stellhorn)

Special purpose computer systems for information retrieval are now being examined in detail. In particular, two designs have been proposed for performing the search and term coordination functions in a document retrieval system. Emphasis is on the use of parallel processing techniques, data base organization and, in some cases, special purpose hardware to permit the searching of large and growing data bases rapidly enough to serve on-line users and to serve large numbers of users simultaneously.

Preliminary attempts to demonstrate the feasibility and potential advantages of these designs have been successful, and a more detailed study program is now beginning. An important feature planned for the new investigation is an analysis of the processing time requirements for the new systems using actual data base characteristics and real search requests from a large, operational, retrieval system.

Publications

- David J. Kuck, "Supercomputers for Ordinary Users," Fall Joint Computer Conference, invited paper, Dec. 1972.
- D. J. Kuck, Y. Muraoka, and S. C. Chen, "On the Number of Operations Simultaneously Executable in FORTRAN-Like Programs and Their Resulting Speed-Up," IEEE Transactions on Computers, Vol. C-21, No. 12, pp. 1293-1310, Dec. 1972.
- D. J. Kuck and A. H. Sameh, "Parallel Computation of Eigenvalues of Real Matrices," Information Processing 71, Vol. II, pp. 1266-1272, North-Holland Publishing Co., Amsterdam-London, 1972.

8. COMPUTER SYSTEMS ANALYSIS

(Supported in part by the National Science Foundation Under Grant No. NSF GJ 28289)

The goal of this research is the development of analytical tools for system modeling and analysis of real time computer networks. Priority assignment and job dispatching rules for a geographically distributed computer network are being investigated.

8.1 Computer Network Modeling (L. Mills and W. McKinney)

Our efforts have been concentrated on developing a viable analytical model for a computer network. A system consisting of a server and a transmitter at each center with exponential service and transmission times has been utilized as the basic unit in the development. Under the assumptions of both infinite and finite queues for the transmitter and the server, these systems have been solved for steady state results and system descriptors have been obtained.

We have also developed separate queueing theory models for the two server and three server systems for the cases in which 1) the servers are assigned jobs randomly, 2) the servers are assigned jobs according to a specified hierarchy, and 3) the servers are assigned jobs on a probabilistic basis. We have shown that the hierarchic scheduling and random scheduling cases were limiting examples of the probabilistic case. Basic expected values were found for the solution for each system.

8.2 Center Throughput Analysis (S. Mamrak and F. Salz)

The GPSS simulation of the IBM 360/75 was used to develop and test a new priority scheme suggested as an alternative to the present system used on the 360. An initial static priority assignment was determined which uses user estimates of CPU, I/O requests and core required by a job. Subsequent priority adjustments are made to reward a job for its long wait in the system or to restore some minimum level of utilization of the CPU and core. Test runs of the new priority scheme on the simulator suggest very substantial improvements in terms of minimizing turnaround time and utilizing system resources.

After the proposed priority scheme was developed, the simulation model was used to evaluate the system performance under each of three conditions:

- 1) Using the existing magic number technique.
- 2) Using the static PRT technique.
- 3) Using a dynamically adjusted PRT.

For each of these tests, an hour of real time was simulated, with identical job streams entering the system. Table I illustrates the results of these tests including O.S. and turnaround times for the job streams, as well as CPU and core utilization values.

In particular, we note the striking decrease in overall turnaround time for jobs processed under the proposed PRT scheduling algorithms. When the resource utilization is kept above some critical level and a maximum waiting time is specified, we observe that the turnaround time for the entire system can, in fact, increase.

TABLE I. SYSTEM PERFORMANCE MEASURES FOR THREE PRIORITY SCHEMES

	PRESENT SCHEME	PRT (Static)	PRT (Dynamic)
Mean HASP Time*			
Class A	100	11	28
B	100	8	9
C	100	98	17
D	-	-	-
A-D	100	12	18
Mean O.S. Time*			
Class A	100	74	94
B	100	54	69
C	100	47	91
D	-	-	-
A-D	100	70	87
Mean Turnaround Time*			
A-D	100	22	29
% CPU Utilization	94	96	98
% CORE Utilization	75	73	73
Total Jobs Processed	482	560	515

*Relative time units (times are normalized to 100 units for each priority class).

Publications

E. K. Bowdon, Sr., and W. J. Barr, "Cost Effective Priority Assignment in Network Computers," Proceedings of the F.J.C.C., December, 1972, pp. 755-763.

9. NUMERICAL ANALYSIS

Additional numerical tests of the adaptive-Chebyshev-factorization algorithm for solving finite element linear systems have been performed. For a mildly complicated triangularization of a rectangular domain, about 65 iterations are required to reduce the relative error to less than 10^{-6} . A proof that the augmented matrix $A+B$ is positive definite has been constructed for certain finite element matrices.

P. E. Saylor

(Supported in part by NSF under Grant No. GJ-31222)

Papers published:

Liu, C. L. and Tseng, C.-C., Complementary Sets of Sequences, IEEE Transactions on Information Theory, Vol. IT-18, No. 5, September 1972.

Liu, C. L., Optimal Scheduling on Multi-Processor Computing Systems, 13th Annual Symposium on Switching and Automata Theory, Maryland, October 1972.

Nievergelt, J., Binary Search Trees and File Organization, Proc. ACM-SIGFIDET Workshop on Data Description, Access and Control, November 1972.

Reingold, E. M., On the Optimality of Some Set Algorithms, J. ACM 19, 649-659, October 1972.

Papers accepted for publication:

Liu, C. L., Analysis and Synthesis of Sorting Algorithms, (to appear in the SIAM Journal on Computing).

Liu, C. L., Ong, B. G. and Ruth, G. R., A Construction Scheme for Linear and Non-Linear Codes, (to appear in the Journal of Discrete Mathematics).

Nievergelt, J. and Reingold, E. M., Binary Search Trees of Bounded Balance, (to appear in SIAM Journal on Computing).

Nievergelt, J., Pradels, J., Wong, C. K. and Yue, P. C., Bounds on the Weighted Path Length of Binary Trees (to appear in Information Processing Letters).

Reingold, E. M., An Efficient List Moving Algorithm, (to appear in Comm. ACM).

Reingold, E. M., Infix to Prefix Translation: The Insufficiency of a Pushdown Stack, (to appear in SIAM J. on Computing).

Wong, C. K., Fuzzy Topology: Product and Quotient Theorems, (to appear in Journal of Math Analysis and Application).

Wong, C. K., On the Optimality of the Probability Ranking Scheme in Storage Applications, (too appear in JACM).

(J. Nievergelt, E. M. Reingold)

11. DISTRIBUTED MINI-COMPUTER COMPUTING SYSTEM (MESH GROUP)

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 36265.)

11.1 MESH Hardware Components

Research was completed on a unique computer structure satisfying many requirements desirable for centralized computer network elements. The research is reported in a masters thesis which will be published as a report next quarter.

The feasibility of designing and fabricating a read-write control memory for the Lockheed SUE computer was investigated.

Memory requirements for the planned research were evaluated and a memory system to meet these needs designed. Procurement of 80K words of core memory in four autonomous modules was completed, and is scheduled for delivery on or about March 1, 1973. Procurement has also been completed for a "2314" type disc which is scheduled for delivery on or about March 15, 1973.

(Mark Ketelsen)

11.2 Memory Segmentation Hardware

A memory segmentation scheme for utilizing a large core memory connected to two buses, and hence many processors and one or more DMA devices was developed. It is the traditional page number, line number and bounds mapping algorithm with a few "special features" thrown in. These are:

1. Overlap of bits used for page and line numbers. This enables larger segment size without reducing the number

of segments at a much lower cost, by using a two-level register set mapping scheme.

2. As the software is likely to be based on the PDP-11 family of machines, segments which are actually "hardware stacks" can grow downwards instead of upwards by using an additional bit in a fashion similar to that of the PDP-11/45 memory segmentation scheme.
3. As different processors may share common segments (or parts of segments) each processor (or DMA device) which may access the memory can have write-protected segments for reading or execution only by unauthorized master devices.
4. As addressable units are bytes and accessible units are words (16 bits) the low order bit is never mapped so that byte/word boundaries are preserved in the transformation.

It is felt that the above features enable the construction of a cheap but flexible segmentation scheme which, at the same time, permits the software to use the memory in an efficient and reliable manner, for regular use as well as interprocessor and inter-bus communication.

(J. Krishnaswamy)

11.3 SUE PDP-11 Emulator

Because the PDP-11 presently being used is on loan from DEC and may be removed at any time it was decided that another PDP-11 be available as a standby. It was also decided that since the PDP-11 is

just a tool for bringing up the network and will not necessarily be part of the network itself it was decided that the PDP-11 should be an emulated PDP-11 microprogrammed on a Lockheed SUE minicomputer thus enabling it to be re-microprogrammed for other use after the network is up. The emulator microprogram is about 75% complete and should be ready to start debugging on the microprogram simulator within the next two weeks. Debugging is expected to take approximately one month by which time a Lockheed SUE with writable control store should be available for final testing.

11.4 CRT Display Terminals

A search was conducted to find an appropriate CRT terminal to use for online debugging of software. Out of the many studied four were selected which met our requirements. They were LSI Model 7700, Beehive Model III, Tee Model 425 and Hazeltine Model 2000A. Of these the Hazeltine was selected because of our familiarity with them and because their extensive use on campus has proven them to be quite reliable. It was decided to rent three of them on a monthly basis.

11.5 PDP-11 Maintenance

In September a PDP-11 with dual DECTape drives and 8K of core was received on loan from DEC. After assembly testing was started and it was discovered that there were several bad gates in the DECTape controller. These gates were replaced and have worked well since. The machine now awaits the addition of the core and disk which are on order.

(Jim Hart)

11.6 MESH Control/Transfer Mechanism (TM Box)

The TM box is an interbus communication facility which will allow several Infibuses or Unibuses to be connected together in a network. The principal operation of this device is to cause an interrupt on one bus when certain events occur on another. Most of the control and synchronization signals of the planned network will go through this device.

The trap operation of the TM box occurs when an address is placed on one of the two buses attached to the TM box and this address is "recognized" by program loadable address decoder. This event causes the status of the significant bus lines to be stored in a set of TM box registers and an interrupt sequence initiated on the other bus. By proper use of this facility a master processor can simulate the presence of any conceivable configuration of device registers for a processor on another bus. This is the principal purpose for which the TM box was designed but there are doubtless many other possible uses for the trap operation.

The transfer operation allows a processor to load a set of TM box registers so as to cause the TM box to become the bus master on the second bus and read or write any location on that bus. This facility allows such things as modifying CPU registers in a slave processor from the control processor.

The operations described above involve many different synchronization problems arising mostly from the fact that an Infibus

may have as many as four CPU's attached to it. These problems and their solutions are presented in the document: TM Box Design Specifications and Operational Characteristics.

A central part of the TM box is the variable address decoder. This unit is being designed so the several different "address range" modules of varying types may be used interchangeably in the event that that proves desirable.

At the present time the TM box logic design is approximately 3/4 complete, the major remaining task being the design of the bus interfaces.

11.7 Microcode Assembler for Lockheed SUE

Since the Lockheed SUE computer is to be used as a central element of the proposed network as a microprogrammable processor several pieces of software will be needed for programming support. The assembler which is being implemented will be written in PDP-11 assembler to run under DOS. The PDP-11 was chosen as the host machine because of the current availability of a useable PDP-11 and the likelihood of most of the SUE's being used to emulate PDP-11's.

The assembler will accept microcode source in symbolic form from several sources, i.e., card, papertape, teletype, etc., and produce in addition to a listing object code on either cards or papertape.

The assembly is done in two passes to allow for symbol processing. Symbolic names may be used for labels, numeric constants or register names. Microinstructions are specified by a series operation specifications which together specify either explicitly or by default the contents of the various fields in the microcode word.

In addition to the instruction generating statement there are also facilities provided to manipulate the location counter, generate the ROM table contents, generate symbols, and format the printed listing.

For complete details of the input specifications see the document: SUE Microcode Assembler Input Specification Manual (11/17/72).

As of this date the assembler is completely written but checkout and debugging have not begun.

(Joe Laprade)

11.8 Lockheed SUE Microcode Simulator

A detailed study of the Lockheed Electronic's SUE processor hardware has been made in order to write a precise simulation program for its microinstructions. This simulation program has been written in PAL-assembler language and is to be run under DOS on the PDP-11. It is an interactive system which supports disk, card reader and line printer I/O, with the user communicates to the system through a TTY. Debugging facilities such as breakpoint, dump and trace are also available on the system to help debugging microprograms. This system has been brought up and test running the SUE "macro-instructions" emulator. Also, a user manual has been written up for this simulation program.

(William Tao)

11.9 MESH Software

11.9.1 System Design

An initial design for a distributed network operating system was developed. This design was based on a configuration which featured dynamic control points and resource pooling. We assumed special purpose microprogrammed minicomputers at the processor node points. As the hardware plans for the network began to evolve, the system design was amended to incorporate a single control processor.

Various compiler building tools were implemented on the PDP-11 in anticipation of an implementation language for the network operating system. A preliminary language reminiscent of PL/1 and PASCAL was brought up for experimentation. Current software development is centered about setting up a software factory for simulating and developing network operating system components. This "factory" consists of adding multiple-access capabilities to an existing single user system. Of these extensions, the CRT multiplexer module and file system eventually will be embedded in the network operating system. However, the essential element of the network operating system, the control structure, must go through another design iteration which depends on the final hardware design.

11.9.2 System Software

Several network system designs were developed. The current plan is based on a graph-like net of specially microprogrammed minicomputers. Control of the net is to be accomplished by a software Hypervisor. Our goals include distributing the control functions and permitting various dynamic reconfigurations of the net to take place. We hope to attain a degree of fail-soft reliability with such techniques.

Current software development is centered about setting up a software factory for simulating and developing Hypervisor components. Progress to this date includes the development of compiler building tools, as PL/1-PASCAL-like implementation language, and a timesharing CRT-Teletype module. Work continues on a file system and a temporary monitor program.

(Gregg Chesson and Tom Miller)

12. GENERAL DEPARTMENT INFORMATION

12.1 Personnel

The number of people associated with the Department in various capacities is given in the following table:

	<u>Full-time</u>	<u>Part-time</u>	<u>FTE</u>
Faculty	21	4	23.27
Visiting Faculty	1	2	2.00
Research Associates and Instructors	0	0	-----
Graduate Research Assistants	0	69	34.75
Graduate Teaching Assistants	0	32	15.75
Professional Personnel	2	1	2.50
Administrative and Clerical	17	0	17.00
Nonacademic Personnel (Monthly)	17	0	17.00
Nonacademic Personnel (Hourly)	0	47	15.10
	<hr/>	<hr/>	<hr/>
TOTAL.....	58	155	117.37

The Department Advisory Committee consists of Professor J. N. Snyder, Head of Department, Professors E. K. Bowdon, D. F. Cudia, M. F. Faiman, H. G. Friedman, C. W. Gear, D. B. Gillies, W. J. Kubitz, D. J. Kuck, C. L. Lui, R. G. Montanelli, S. Muroga, T. A. Murrell, J. Nievergelt, J. R. Phillips, W. J. Poppelbaum, S. R. Ray, E. M. Reingold, J. E. Robertson, P. E. Saylor, D. L. Slotnick, J. E. Vander Mey, D. S. Watanabe, and T. Wilcox.

12.2 Bibliography

During the fourth quarter, the following publications were issued by the laboratory:

Report Numbers

No. 544 Phillips, J. Richard, "The Structure and Design Philosophy of OL/2 - An Array Language - Part I: Language Overview," October, 1972.

Theses

No. 532 Verma, Shiv Prakash, "Perspective Transformer for the Stereomatrix 3-D Display System," October, 1972. (Ph.D. Thesis).

No. 533 Maruyama, Kiyoshi, "A Study of Visual Shape Perception," October, 1972 (Ph.D. Thesis).

No. 543 Mora-Tovar, Jose Joaquin, "A Study of the Effect of Additional Inequalities in Integer Programming for Logical Design," October, 1972 (M.S. Thesis).

No. 545 Chin, Janet Sau-Ying, "Scheduling on Parallel Processors for Weighted-Node Graphs," October, 1972. (M.S. Thesis).

No. 546 Larsen, Leonard Andrew, "Automatic Solutions for Partial Differential Equations," October, 1972. (Ph.D. Thesis).

No. 548 McInnes, Allan William, "On the Uniform Approximation of a Class of Singular Integral Equations in a Holder Space," December, 1972 (Ph.D. Thesis).

No. 549 Trout, Harold Robert George, "Parallel Techniques," October, 1972 (Ph.D. Thesis).

No. 552 Lermitt, Raymond Jonathon, "Numerical Methods for the Identification of Differential Equations," October, 1972 (Also CAC Document #49) (Ph.D. Thesis).

No. 553 Mercer, Robert L., "Partial Wave Analysis of Elastic and Inelastic Scattering of Dirac Particles," October, 1972 (Ph.D. Thesis).

No. 554 des Trois Maisons, Paul E. N., "Some Proposals for Modifying the GRASS Language," December, 1972 (M.S. Thesis).

Theses Cont'd

No. 558

Stevens, James E., Jr., "Fast Heuristic Techniques for Placing and Wiring Printed Circuit Boards," October, 1972 (Also CAC Document #58) (Ph.D. Thesis).

File Numbers

No. 873

Borovec, Richard T. and Lewis, George T., "Image 8: Maintenance Processor/Pau Interface," November, 1972.

Borovec, R. T. and Lewis, G. T., "IMAGE 8: Maintenance Processor/PAU Interface," UIUCDCS-F-873, Department of Computer Science, University of Illinois, Urbana, Illinois, November 1972.

Abstract:

This document is intended to serve as the programming and maintenance manual for the Maintenance Processor/PAU interface.

Bowdon, E. K., Sr., and Barr, W. J., "Cost Effective Priority Assignment in Network Computers," Proceedings of the F.J.C.C., December, 1972, pp. 755-763.

Abstract:

With the advent of network computers, a new area of computer systems analysis has evolved. Unfortunately, most of the work which has been done to date merely extends the previously existing theory of communications. While this work has been very fruitful and produced important results, our analysis is predicated on the assumption that a geographically distributed network computer is, in reality, quite different from telephone networks and individual computing centers.

In this paper, we focus our attention on the probable goals of the networks and define a measure of cost effectiveness. Then using this measure we develop a priority assignment technique for the individual centers that comprise the network. We conclude by expanding the measure of cost effectiveness to determine load leveling rules for the entire network.

des Trois Maisons, Paul E. N., "Some Proposals for Modifying the GRASS Graphics Language," University of Illinois at Urbana-Champaign, Urbana, Illinois, UIUCDCS-R-72-554, December 1972.

Abstract:

The purpose of this research is to outline a reconfiguration of the Graphical Remote Access Support System (GRASS) as developed at the University of Illinois. In this paper, we shall attempt to improve upon the graphical procedures of GRASS, while still fulfilling the original system goals. For this reason, familiarity with the current system is required.

Gear, C. W., "A Generalized Interactive Network Analysis and Simulation System," USA-Japan Computer Conference, Tokyo, Japan, October 3-5, 1972.

Abstract:

The purpose of a package being developed at the University of Illinois at Urbana is to make the simulation of models from great many disciplines possible within a single framework, so that new concepts in the system can be incorporated into the model with ease, and so that interdisciplinary models can be handled directly (e.g., a biological model connected to an electrical network).

Gear, C. W., "The Effect of Variable Mesh Size on the Stability of Multistep Methods," Conference on Ordinary Differential Equations, Austin, Texas, October 19-20, 1972.

Abstract:

The Effect of two different schemes for implementing variable mesh sizes in multistep methods are investigated. It is proved that one is more stable than the other for some cases, but that both are stable when the step changes are small. The practical implications of these results are discussed.

A model consists of a number of interconnected elements. Initially, the user must specify the behavior of elements he will use, and say how they can be connected.

The development and simulation of a model goes through three distinct phases: an input phase in which graphical and textual information are manipulated, a symbolic phase in which the questions are manipulated and finally compiled into machine language substituted, and a numerical in which the behavior of the model is determined.

Kuck, David J., "Supercomputers for Ordinary Users," Fall Joint Computer Conference, invited paper, December 1972.

Abstract:

Today's minicomputers were the supercomputers of a few generations ago. Will trend continue? Present analysis show that a large number of operations may be performed simultaneously in executing ordinary FORTRAN programs as well as programs in some other languages quite removed from FORTRAN. In the future it may be possible to build fast computers from slow parts, by performing many operations concurrently.

This paper sketches some recent results in the analysis of programs and how to use these results in the design of computer systems. The speed and structure of resulting machine organizations are discussed in terms of accessing, aligning and processing data.

Liu, C. L., "Optimal Scheduling on Multi-Processor Computing Systems,"
13th Annual Symposium on Switching and Automata Theory,
Maryland, October 1972.

Abstract:

A set of tasks $\{T_1, T_2, \dots, T_n\}$ are to be scheduled on a two-processor computing system. The execution time of each of the tasks is given. Moreover, a partial ordering $<$ over the set of tasks is specified. If $T_i < T_j$, it is required that the execution of T_j will not begin until the completion of the execution of T_i . Let ω denote the total elapsed time for the execution of all the tasks in the set when an optimal non-preemptive schedule is used. Let ω' denote the total elapsed time for the execution of all the tasks in the set when an optimal preemptive schedule is used. Trivially, we know that

$$\omega \geq \omega'$$

However, it can be shown that

$$\omega' \geq \frac{3}{4} \omega$$

This result can be extended to an n -processor computing system. In this case,

$$\omega' \geq \frac{n+1}{2n} \omega$$

Moreover, these are best possible bounds.

A possible interpretation of the results is: The installation of a high speed drum in a two-processor computing system will increase the operation speed of the system by at most 25%. For an n -processor system, the increment is at most 50%. (The installation of a high speed drum enables us to interrupt the execution of tasks at will).

Liu, C. L. and Tseng, C. C., "Complementary Sets of Sequence," IEEE Transactions on Information Theory, Vol. IT-18, No. 5, September 1972.

Abstract:

A set of equally long finite sequences, the elements of which are either +1 or -1, is said to be a complementary set of sequences if the sum of autocorrelation functions of the sequences in that set is zero except for a zero-shift term. A complementary set of sequences is said to be a mate of another set if the sum of the cross-correlation functions of the corresponding sequences in these two sets is zero everywhere. Complementary sets of

Kuck, David J., Muraoka, and Chen, S. C., "On the Number of Operations Simultaneously Executable in FORTRAN-Like Programs and Their Resulting Speed-Up," IEEE Transactions on Computers, Vol. C-21, No. 12, pp. 1293-1310, December 1972.

Abstract:

This paper is concerned with the problem of analyzing ordinary FORTRAN-like programs to determine how many of their operations could be performed simultaneously. Algorithms are presented for handling arithmetic assignment statements, DO loops and IF statement trees. The height of the parse trees of arithmetic expressions are reduced by distribution of multiplication over addition as well as the use of associativity and commutativity. DO loops are analyzed in terms of their index sets and subscript forms. Some general undelying assumptions about machine organization are also given. In terms of several measures which are defined, the results of experimental analyses are presented. About 20 FORTRAN IV programs consisting of nearly 1000 source cards were analyzed. Evidence is given that for very simple FORTRAN programs 16 processors could be effectively used operating simultaneously in a parallel or pipeline fashion. Thus, for medium or large size FORTRAN programs, machines consisting of multiples of a basic 16 processor unit could be used.

Kuck, D. J. and Sameh, A. H., "Parallel Computation of Eigenvalues of Real Matrices," Information Processing 71, Vol. II, pp. 1266-1272, North-Holland Publishing Co., Amsterdam-London, 1972.

Abstract:

To realize the full speed potential of forthcoming array computers, one must properly organize both an algorithm for the arithmetic unit and the data in a highly interleaved memory. We consider matrix eigenvalue calculations by the Jacobi, Householder and QR methods. In each case we present storage schemes for the matrices in a parallel memory which allows simultaneous access to proper elements. In terms of these storage schemes we discuss parallel implementations of the above mentioned algorithms and compute overall efficiencies of machine use.

Larsen, Leonard Andrew, "Automatic Solutions of Partial Differential Equations. UIUCDCS-R-72-546, University of Illinois at Urbana-Champaign, Urbana, Illinois, October 1972.

Abstract:

The problem of the present paper is to look for an automatic method that can be applied to some subset of partial differential

sequences are said to be mutually orthogonal complementary sets if any two of them are mates to each other. In this paper we discuss the properties of such complementary sets of sequences. Algorithms for synthesizing new sets from a given set are given. Recursive formulas for constructing mutually orthogonal complementary sets are presented. It is shown that matrices consisting of mutually orthogonal complementary sets of sequences can be used as operators so as to perform transformations and the Hadamard transformation suggests applications of such new transformations to signal processing and image coding.

Maruyama, K., "A Study of Visual Shape Perception," UIUCDCS-R-72-533, Department of Computer Science, University of Illinois, Urbana, Illinois, October 1972.

Abstract:

With the use of computers, this paper develops and studies algorithms involved in visual shape perception. The main interest is to solve the naming problem, i.e., the strategies by which map-like information is transferred to the scene under analysis.

Nievergelt, J., "Binary Search Trees and File Organization," Proc. ACM-SIGFIDET Workshop on Data Description, Access and Control, November 1972.

Abstract:

Binary search trees are an important technique for organizing large files, because they are efficient for both random and sequential access of records, and for modification of a file. Because of this, they have received a great deal of attention in recent years, and their properties are now better understood than those of most other file organization methods. This paper surveys the main results which have been obtained.

Read, J. S., "Parallel Image-Processing for Automated Cervical Smear Analysis," UIUCDCS-R-72-497, Department of Computer Science, Urbana, Illinois, October 1972.

Abstract:

The prime objective of this work was to examine the effectiveness of a parallel digital image processor in the analysis of cervical (Pap) smear imagery. This task has historically been shown to be very difficult to do by machine, partly because clumping and overlapping of cells has presented difficulties exceeding the capacity of cost-effective image-processing technology. The

ILLIAC III's special parallel image processor promises a capability to analyze much more complex patterns for a given cost than has been possible in the past.

This thesis contains background information on cervical smears, an extensive review of automated cytology, and reports on three experiments: In (1) detecting enlarged cell nuclei in the presence of clumps of leukocytes, (2) detecting super-textures occurring in cervical preparations. The last experiment utilized the Varivalued logic approach to pattern recognition developed by McCormick and Michalski.

Reingold, E. M., "On the Optimality of Some Set Algorithms," J. ACM 19, 649-659, October 1972.

Abstract:

The establishment of lower bounds on the number of comparisons necessary to solve various combinatorial problems is considered. Some of the new results are: (a) given two finite sets of real numbers, A and B, where $n = \max(|A|, |B|)$, $O(n \log n)$ comparisons are required to determine if $A = B$, even when comparisons are allowed between linear functions of the numbers; and (b) the maximum of a set of n real numbers cannot be computed in fewer than $n - 1$ comparisons if comparisons of only linear functions of the numbers are permitted, but the maximum can be computed in $\lceil \log_2 n \rceil$ comparisons if comparisons are allowed between exponential functions of the numbers.

Verma, Shiv Prakash, "Perspective Transformer for the Stereomatrix 3-D Display System," Department of Computer Science Report No. 532, University of Illinois, Urbana-Champaign, October 1972.

Abstract:

STEREOMATRIX is a large screen interactive 3-D display system which presents computer-generated transparent "wire-frame" drawings stereoscopically. A 3-D cross called the cursor can be moved around in the viewing space by the observer by means of a joystick. This feature allows the operator to select a point in the viewing space for graphic manipulation and relay it to the computer space. He can also use the joystick to draw pictures in the 3-D viewing space. The perspective of the figure changes with observer movement in such a way that the figure appears to be stationary in space.

"APL in Exposition," by Dr. K. E. Iverson, International Business Machines Corporation, Philadelphia Scientific Center, 3401 Market Street, Philadelphia, Pennsylvania, October 2, 1972.

"Automating Our Introductory Computer Science Courses," by Professor Jurg Nievergelt, Department of Computer Science, University of Illinois, Urbana, Illinois, October 23, 1972.

"Two-Server, Two-Class System: Application to a Multiprogrammed Computer," by Dr. Vijay P. Marathe, Department of Business Administration, 350 Commerce West, University of Illinois, Champaign, Illinois, October 30, 1972.

"Artificial Intelligence as a New Theory of Education," by Professor Seymour A. Papert, The Artificial Intelligence Laboratory, 545 Technology Square, Cambridge, Massachusetts, November 6, 1972.

"On the Optimality of the Probability Ranking Scheme in Storage Application," by Professor C. K. Wong, Department of Computer Science, University of Illinois, Urbana, Illinois, November 13, 1972.

"Fast Computers From Slow Parts," by Professor David Kuck, Department of Computer Science, University of Illinois, Urbana, Illinois, November 20, 1972.

"Microprogramming and Directly Executable Languages," by Professor Michael J. Flynn, Department of Computer Science, The Johns Hopkins University, Baltimore, Maryland, December 4, 1972.

"Tree Polynomials," by Professor Murray Edelberg, Department of Electrical Engineering, Princeton University, Princeton, New Jersey, December 18, 1972.

12.5 Drafting

During the fourth quarter, a total of 325 drawings were processed by the general departmental drafting section:

Formal Drawings

Large Drawings	54
Medium Drawings	173
Small Drawings	40
Layout	1
Report Drawings	29
Change Order Drawings	0
Miscellaneous Drawings	28
Completed Total Drawings.	325

(M. Goebel)

12.6 Shop's Production

Job orders processed and completed during the fourth quarter of 1972 are as follows:

	<u>AEC 2118</u>	<u>AEC 1469</u>	<u>Other</u>
Machine Shop	0	6	2
Electronic Shop	6	43	+1
Chemical Shop	5	35	8
Layout Shop	25	29	2

(F. P. Serio)

DEPARTMENT OF COMPUTER SCIENCE
GRADUATE COLLEGE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

REPORT REQUEST FORM

Report Number

Title

Fold and Staple as shown

NAME:

ADDRESS:

Fill out Mailing Label



NAME:

ADDRESS:

CUT ALONG THIS LINE

STAPLE

FOLD

Mail Room
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

Stamp

FOLD

BIOGRAPHIC DATA HEET		1. Report No. UIUCDCS-QPR-72-4	2.	3. Recipient's Accession No.
Title and Subtitle Quarterly Progress Report			5. Report Date	
Author(s)			6.	
Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801			8. Performing Organization Rept. No.	
Sponsoring Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801			10. Project/Task/Work Unit No.	
			11. Contract/Grant No.	
			13. Type of Report & Period Covered Quarterly Progress Report Oct - Nov - Dec	
			14.	
Supplementary Notes				
Abstracts Not Applicable				
Key Words and Document Analysis, 17a. Descriptors Not Applicable				
b. Identifiers/Open-Ended Terms Not Applicable				
c. COSATI Field/Group				
Availability Statement RELEASE UNLIMITED			19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 148
			20. Security Class (This Page) UNCLASSIFIED	22. Price



UNIVERSITY OF ILLINOIS-URBANA



3 0112 084228235